A methodology for the rigorous verification of plasma simulation codes

Fabio Riva[1,2]

P. Ricci[1], C. Beadle[1], F.D. Halpern[1], S. Jolliet[1],
J. Loizu[1], J. Morales[1], A. Mosetto[1], P. Paruta[1], C. Wersal[1]

[1]École Polytechnique Fédérale de Lausanne (EPFL), Swiss Plasma Center (SPC), Switzerland
[2]CCFE, Culham Science Centre, Abingdon, Oxon, OX14 3DB, United Kingdom

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

SWISS PLASMA CENTER

CCFE CULHAM CENTRE FOR FUSION ENERGY

Verification Procedures

How to rigorously ensure that a simulation code is bug-free?

Code Verification

How to estimate the numerical uncertainty affecting simulation results?

Solution Verification

Code Verification Techniques

1. Simple tests

   Energy conservation, convergence (without a known exact solution)

2. Code-to-code comparison (benchmarking)

   Example: Cyclone test case

NOT RIGOROUS CODE VERIFICATION

3. Convergence tests

   Do results converge to the exact solution?

4. Order of accuracy tests

   Do results converge to the exact solution at the expected rate?

RIGOROUS CODE VERIFICATION, but require analytical solution

Code Verification Techniques

1. Simple tests

   Energy conservation, convergence (<u>without</u> a known <u>exact solution</u>)

2. Code-to-code comparison (benchmarking)

   Example: Cyclone test case

NOT RIGOROUS CODE VERIFICATION

3. Convergence tests

   Do results converge to the <u>exact solution</u>?

4. Order of accuracy tests

   Do results converge to the <u>exact solution</u> at the expected rate?

RIGOROUS CODE VERIFICATION, but require analytical solution

The only procedure ensuring both convergence and correct numerical implementation

Order of accuracy test

Model: $M(s) = 0$

Solve $M_h(s_h) = 0$, with h discretization parameter

Compute the numerical error $\epsilon_h = \|s - s_h\|$

Order of accuracy test

Model: $M(s) = 0$

Solve $M_h(s_h) = 0$, with h discretization parameter

Compute the numerical error $\epsilon_h = \|s - s_h\|$

If $\epsilon_h = Ch^p + \mathcal{O}\left(h^{p+1}\right)$, with p the order of accuracy, the code is verified

Order of accuracy test

Model: $M(s) = 0$

Solve $M_h(s_h) = 0$, with h discretization parameter

Compute the numerical error $\epsilon_h = \|s - s_h\|$

If $\epsilon_h = Ch^p + \mathcal{O}\left(h^{p+1}\right)$, with p the order of accuracy, the code is verified

Model: $M(s) = 0$, s unknown

Solve $M_h(s_h) = 0$ but $\epsilon_h = \|s - s_h\| = ?$

Method of Manufactured Solutions

MMS developed by CFD community for verifying codes based on finite difference schemes

[Roache *et al.*, AIAA J. (1984); Oberkampf *et al.*, AIAA J. (1998)]

Method of Manufactured Solutions (MMS):

1. Choose s_m and compute $S = M(s_m)$
2. Define $G = M - S$, $G(s_m) = 0$
3. Solve $G_h(s_{m,h}) = M_h(s_{m,h}) - S = 0$
4. Obtain $\epsilon_h = \|s_m - s_{m,h}\|$

Method of Manufactured Solutions

MMS developed by CFD community for verifying codes based on finite difference schemes

[Roache *et al.*, AIAA J. (1984); Oberkampf *et al.*, AIAA J. (1998)]

Method of Manufactured Solutions (MMS):

1. Choose s_m and compute $S = M(s_m)$
2. Define $G = M - S$, $G(s_m) = 0$
3. Solve $G_h(s_{m,h}) = M_h(s_{m,h}) - S = 0$
4. Obtain $\epsilon_h = \|s_m - s_{m,h}\|$

If $\epsilon_h = C' h^p + \mathcal{O}\left(h^{p+1}\right)$ the code is verified

Method of Manufactured Solutions

MMS developed by CFD community for verifying codes based on finite difference schemes

[Roache *et al.*, AIAA J. (1984); Oberkampf *et al.*, AIAA J. (1998)]

<u>Method of Manufactured Solutions (MMS):</u>

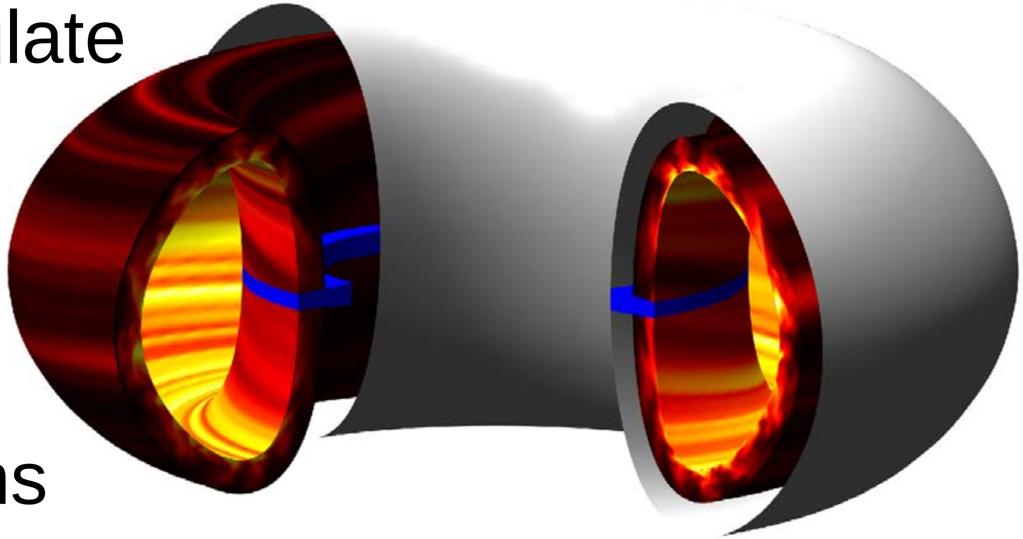
1. Choose s_m and compute $S = M(s_m)$
2. Define $G = M - S$, $G(s_m) = 0$
3. Solve $G_h(s_{m,h}) = M_h(s_{m,h}) - S = 0$
4. Obtain $\epsilon_h = \|s_m - s_{m,h}\|$

While arbitrary, s_m should excite all terms in equations and ensure no dominating component in numerical error

First MMS plasma simulation code verification

GBS: 3D fluid code used to simulate SOL plasma turbulence

[Ricci *et al.*, PPCF (2012)]



Drift-reduced Braginskii equations

- Continuity equation: $\frac{dn}{dt} = \nabla_{\parallel} \cdot \left(n v_{\parallel e} \mathbf{b} \right) + \ldots$

Numerical scheme:

- RK4 for time integration
- 2nd order finite differences for spatial derivatives
- Arakawa scheme for $E \times B$ advection terms

$$\Rightarrow \epsilon_h = \mathcal{O}(\Delta x^2) + \mathcal{O}(\Delta y^2) + \mathcal{O}(\Delta z^2) + \mathcal{O}(\Delta t^4)$$

Verification of GBS [Riva *et al.*, PoP (2014)]

- Choose $s_m(x, y, z, t) = A[B + C\sin(Dy)\sin(Ex + ...$
- Compute $S = M(s_m)$
- Choose $\Delta x_0,\ \Delta y_0,\ \Delta z_0,\ \Delta t_0$
- Define

$$h = \frac{\Delta x}{\Delta x_0} = \frac{\Delta y}{\Delta y_0} = \frac{\Delta z}{\Delta z_0} = \left(\frac{\Delta t}{\Delta t_0}\right)^2$$

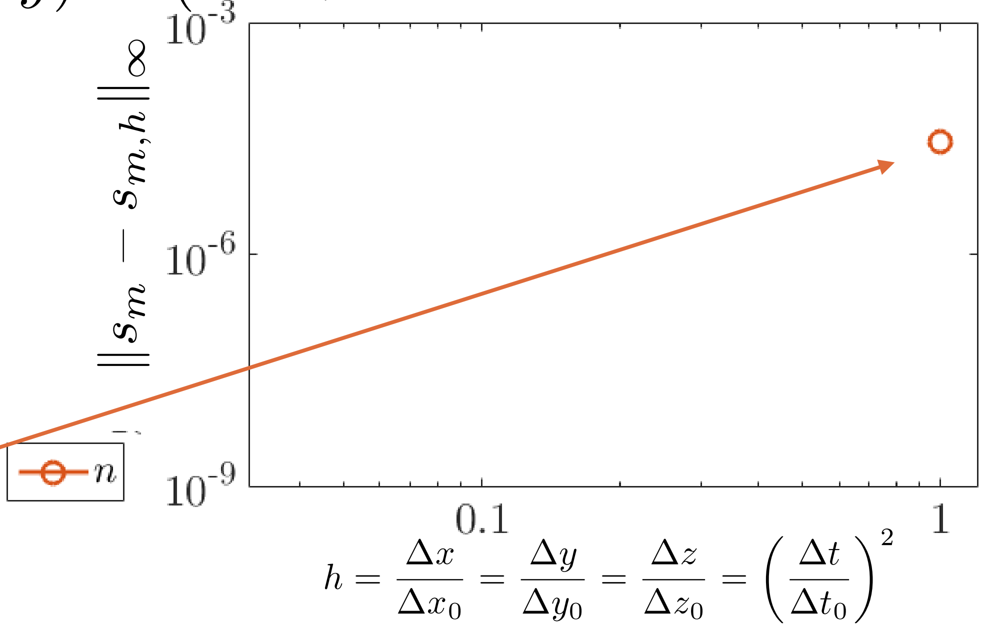
- Obtain $s_{m,h}$ for $h = 1$
- Compute $\epsilon_h = \|s_m - s_{m,h}\|$

Verification of GBS [Riva *et al.*, PoP (2014)]

- Choose $s_m(x, y, z, t) = A[B + C\sin(Dy)\sin(Ex + \ldots$
- Compute $S = M(s_m)$
- Choose $\Delta x_0,\ \Delta y_0,\ \Delta z_0,\ \Delta t_0$
- Define

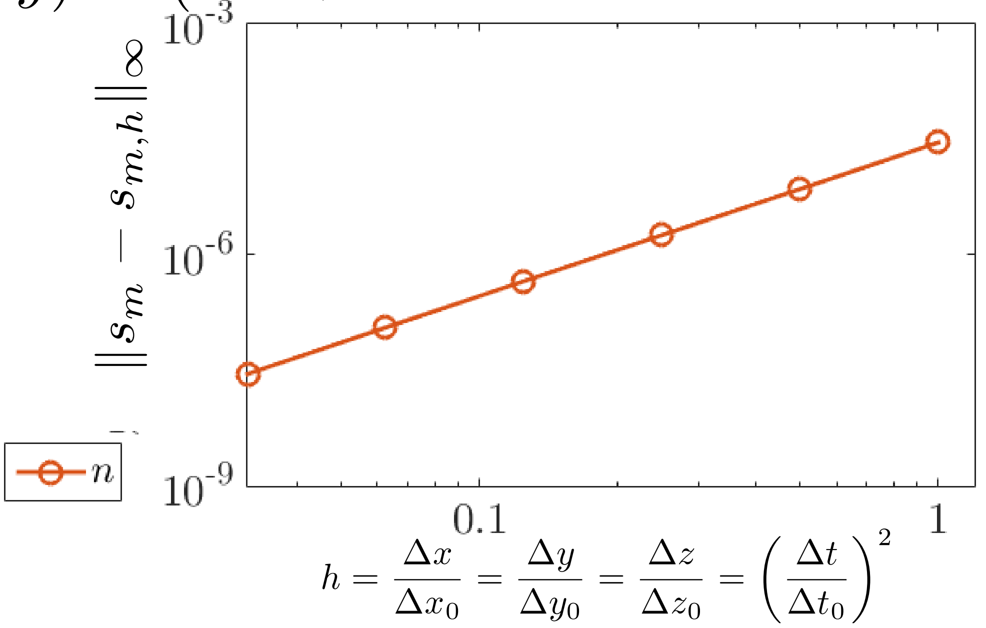
$$h = \frac{\Delta x}{\Delta x_0} = \frac{\Delta y}{\Delta y_0} = \frac{\Delta z}{\Delta z_0} = \left(\frac{\Delta t}{\Delta t_0}\right)^2$$

- Obtain $s_{m,h}$ for $h = 1$
- Compute $\epsilon_h = \|s_m - s_{m,h}\|$



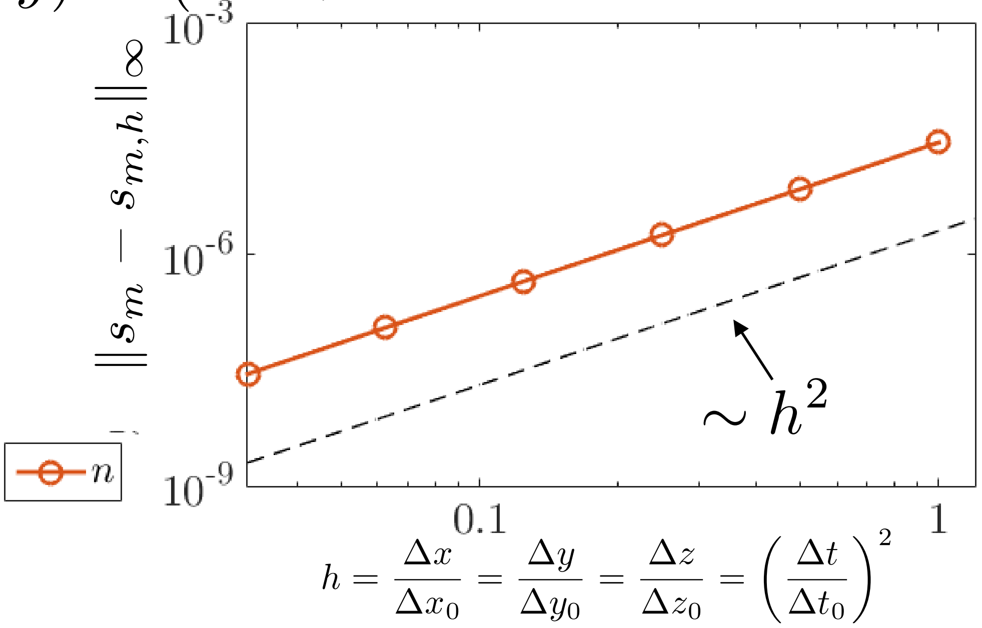
Verification of GBS [Riva *et al.*, PoP (2014)]

- Choose $s_m(x,y,z,t) = A[B + C\sin(Dy)\sin(Ex + \ldots$
- Compute $S = M(s_m)$
- Choose $\Delta x_0,\ \Delta y_0,\ \Delta z_0,\ \Delta t_0$
- Define
$$h = \frac{\Delta x}{\Delta x_0} = \frac{\Delta y}{\Delta y_0} = \frac{\Delta z}{\Delta z_0} = \left(\frac{\Delta t}{\Delta t_0}\right)^2$$
- Obtain $s_{m,h}$ for $h = 1$
- Compute $\epsilon_h = \|s_m - s_{m,h}\|$
- Refine the grid
- Obtain $s_{m,h}$ for $h < 1$



Verification of GBS [Riva *et al.*, PoP (2014)]

- Choose $s_m(x, y, z, t) = A[B + C\sin(Dy)\sin(Ex + \ldots$
- Compute $S = M(s_m)$
- Choose $\Delta x_0,\ \Delta y_0,\ \Delta z_0,\ \Delta t_0$
- Define
$$h = \frac{\Delta x}{\Delta x_0} = \frac{\Delta y}{\Delta y_0} = \frac{\Delta z}{\Delta z_0} = \left(\frac{\Delta t}{\Delta t_0}\right)^2$$
- Obtain $s_{m,h}$ for $h = 1$
- Compute $\epsilon_h = \|s_m - s_{m,h}\|$
- Refine the grid
- Obtain $s_{m,h}$ for $h < 1$
- Verify that $\epsilon_h = Ch^2 + \mathcal{O}(h^3)$



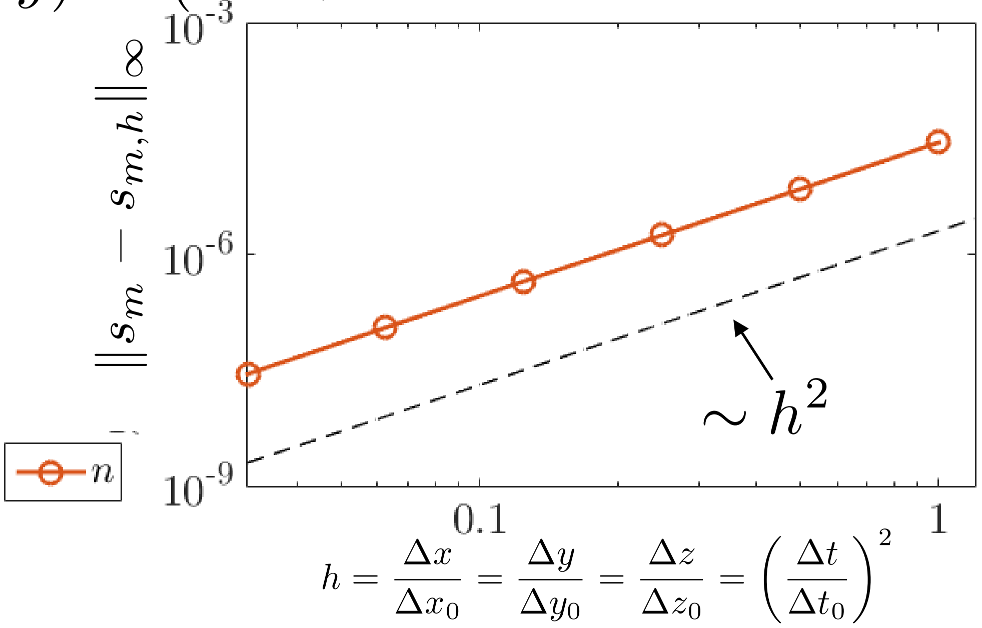
Verification of GBS [Riva *et al.*, PoP (2014)]

- Choose $s_m(x, y, z, t) = A[B + C\sin(Dy)\sin(Ex + \ldots$
- Compute $S = M(s_m)$
- Choose $\Delta x_0,\ \Delta y_0,\ \Delta z_0,\ \Delta t_0$
- Define

$$h = \frac{\Delta x}{\Delta x_0} = \frac{\Delta y}{\Delta y_0} = \frac{\Delta z}{\Delta z_0} = \left(\frac{\Delta t}{\Delta t_0}\right)^2$$

- Obtain $s_{m,h}$ for $h = 1$
- Compute $\epsilon_h = \|s_m - s_{m,h}\|$
- Refine the grid
- Obtain $s_{m,h}$ for $h < 1$
- Verify that $\epsilon_h = Ch^2 + \mathcal{O}(h^3)$
- Define the observed order of accuracy

$$\hat{p} = \ln\left(\frac{\epsilon_{rh}}{\epsilon_h}\right) / \ln(r)$$



Verification of GBS [Riva *et al.*, PoP (2014)]

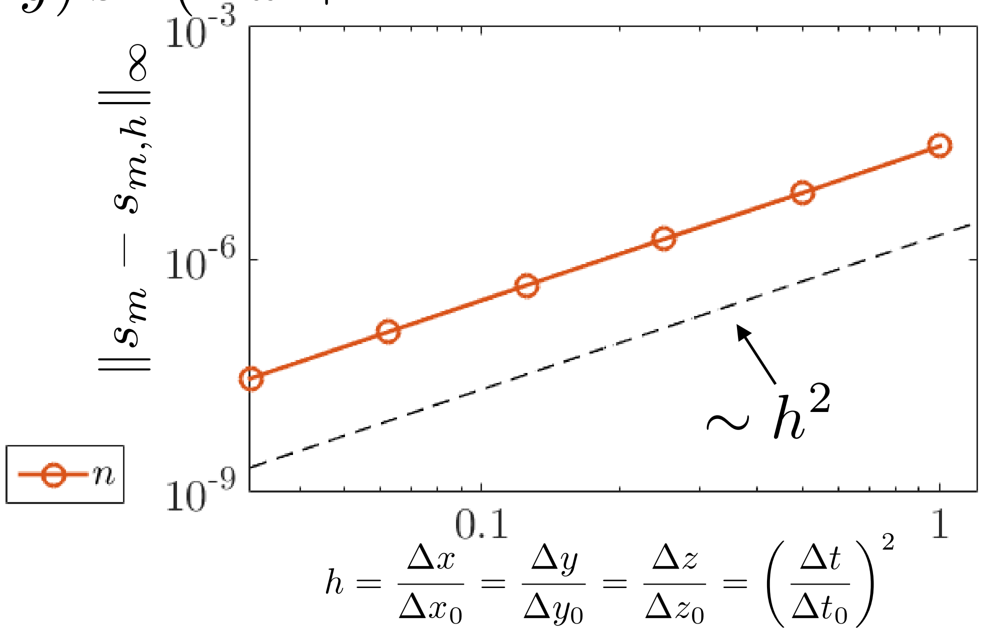
- Choose $s_m(x, y, z, t) = A[B + C\sin(Dy)\sin(Ex + \ldots$
- Compute $S = M(s_m)$
- Choose $\Delta x_0,\ \Delta y_0,\ \Delta z_0,\ \Delta t_0$
- Define

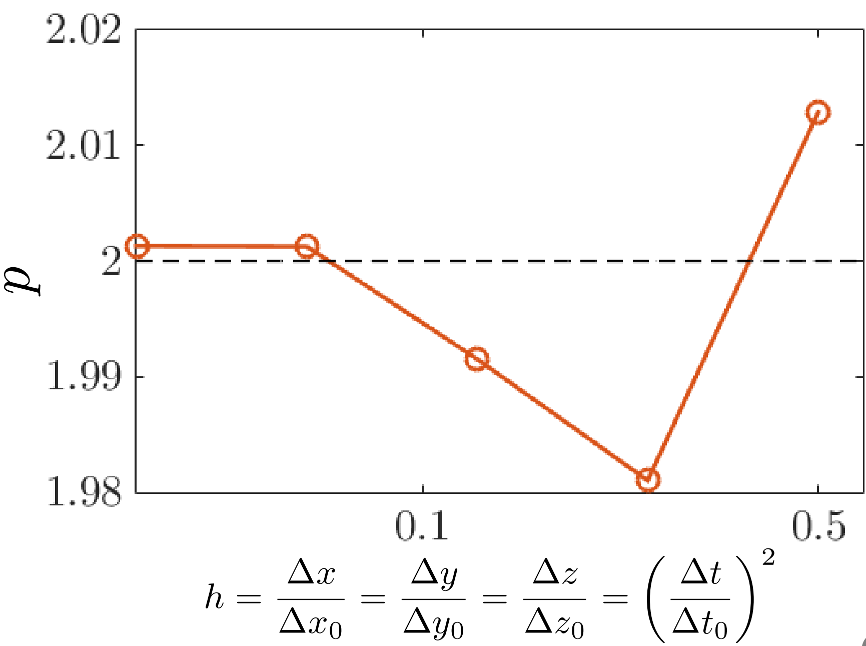
$$h = \frac{\Delta x}{\Delta x_0} = \frac{\Delta y}{\Delta y_0} = \frac{\Delta z}{\Delta z_0} = \left(\frac{\Delta t}{\Delta t_0}\right)^2$$

- Obtain $s_{m,h}$ for $h = 1$
- Compute $\epsilon_h = \|s_m - s_{m,h}\|$
- Refine the grid
- Obtain $s_{m,h}$ for $h < 1$
- Verify that $\epsilon_h = Ch^2 + \mathcal{O}(h^3)$
- Define the observed order of accuracy

$$\hat{p} = \ln\left(\frac{\epsilon_{rh}}{\epsilon_h}\right) / \ln(r)$$

- Verify that $\hat{p} \to p$





Verification of GBS [Riva *et al.*, PoP (2014)]

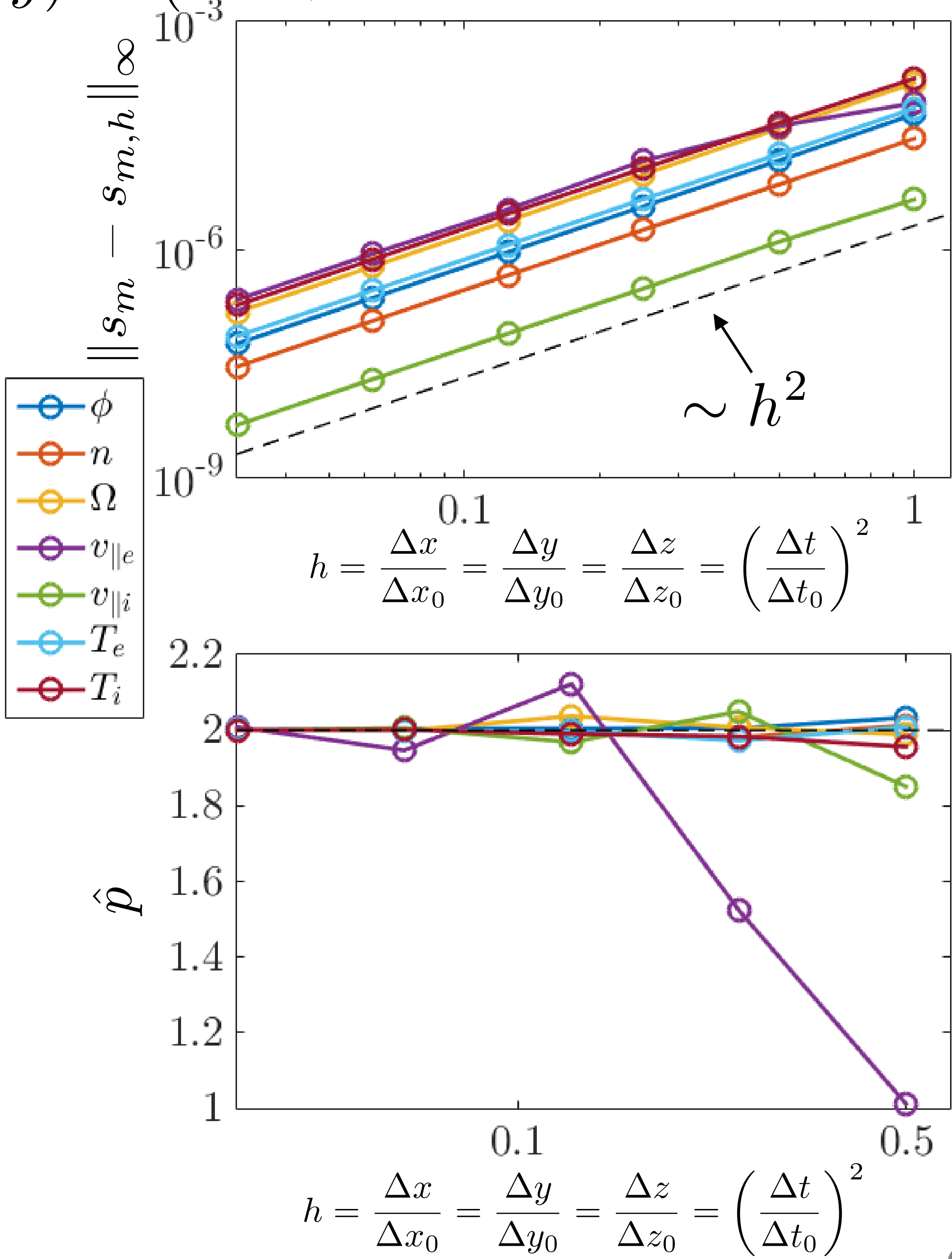
- Choose $s_m(x,y,z,t) = A[B + C\sin(Dy)\sin(Ex + \ldots$
- Compute $S = M(s_m)$
- Choose $\Delta x_0,\ \Delta y_0,\ \Delta z_0,\ \Delta t_0$
- Define

$$h = \frac{\Delta x}{\Delta x_0} = \frac{\Delta y}{\Delta y_0} = \frac{\Delta z}{\Delta z_0} = \left(\frac{\Delta t}{\Delta t_0}\right)^2$$

- Obtain $s_{m,h}$ for $h = 1$
- Compute $\epsilon_h = \|s_m - s_{m,h}\|$
- Refine the grid
- Obtain $s_{m,h}$ for $h < 1$
- Verify that $\epsilon_h = Ch^2 + \mathcal{O}(h^3)$
- Define the observed order of accuracy

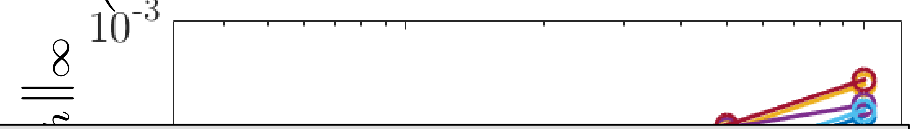
$$\hat{p} = \ln\left(\frac{\epsilon_{rh}}{\epsilon_h}\right) / \ln(r)$$

- Verify that $\hat{p} \to p$



Verification of GBS [Riva *et al.*, PoP (2014)]

- Choose $s_m(x, y, z, t) = A[B + C\sin(Dy)\sin(Ex + ...$
- Compute $S = M(s_m)$



GBS is verified!

First application of MMS for the verification of a plasma simulation code based on finite difference schemes

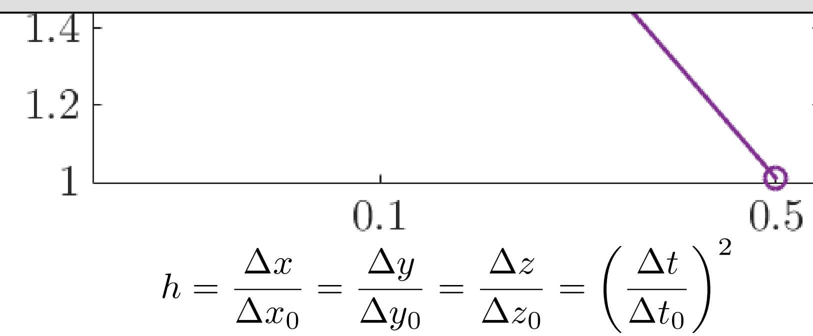
[Riva *et al.*, PoP (2014)]

MMS now routinely used to verify plasma turbulence codes

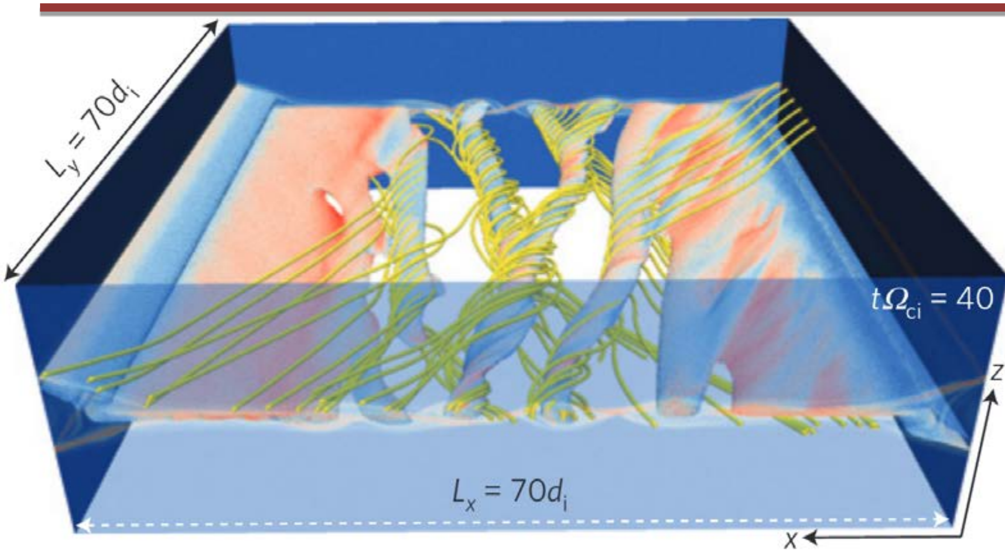
[Tamain *et al.*, JCP (2016); Dudson *et al.*, PoP (2016);…]

$$\hat{p} = \ln\left(\frac{\epsilon_{rh}}{\epsilon_h}\right) / \ln(r)$$

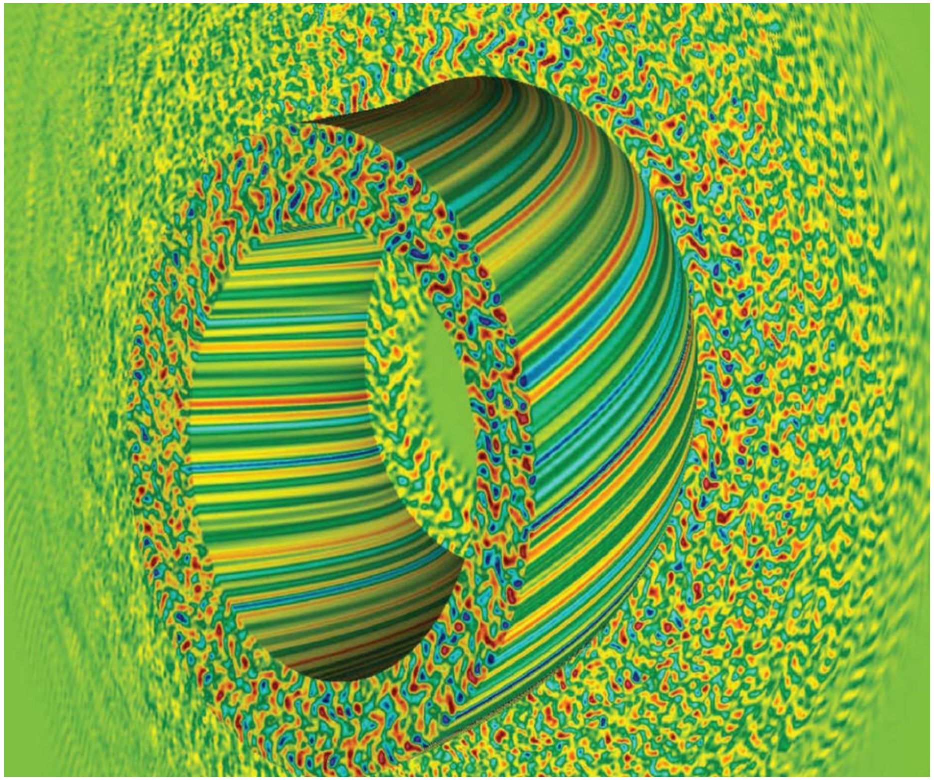
- Verify that $\hat{p} \to p$



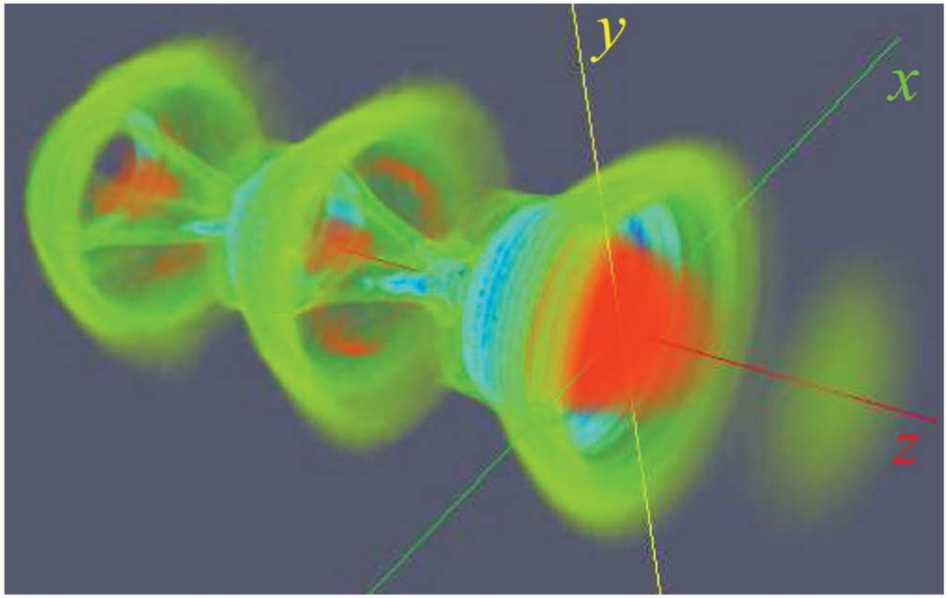
Particle-In-Cell (PIC) codes



[Daughton *et al.*, Nature (2011)]



[Fasoli *et al.*, Nature (2016)]



[Gordon *et al.*, PRL (2008)]

The PIC algorithm

A simple model:

$$\frac{\partial f}{\partial t} + v \cdot \frac{\partial f}{\partial x} + \frac{q}{m} E \cdot \frac{\partial f}{\partial v} = 0$$

$$\frac{\partial E}{\partial x} = \frac{\rho}{\epsilon_0}$$

The PIC algorithm

A simple model:

$$\frac{\partial f}{\partial t} + v \cdot \frac{\partial f}{\partial x} + \frac{q}{m} E \cdot \frac{\partial f}{\partial v} = 0$$

$$\frac{\partial E}{\partial x} = \frac{\rho}{\epsilon_0}$$

Introduce N markers (superparticles) and approximate

$$f \approx f_N(x, v, t) = \sum_{p=1}^{N} \delta\left[x - x_p(t)\right] \delta\left[v - v_p(t)\right]$$

with $x_p,\ v_p$ satisfying equations of motion

MMS for a PIC simulation code

The modified model: $$\frac{\partial f_m}{\partial t} + v \cdot \frac{\partial f_m}{\partial x} + \frac{q}{m} E_m \cdot \frac{\partial f_m}{\partial v} = S_f$$

$$\frac{\partial E_m}{\partial x} = \frac{\rho}{\epsilon_0} + S_E$$

$$f_m \approx f_N(x, v, t) = \sum_{p=1}^{N} w_p(t) \delta\left[x - x_p(t)\right] \delta\left[v - v_p(t)\right]$$

with $$\frac{dw_p}{dt} = \frac{S_f[x_p(t), v_p(t), t]}{f_m[x_p(0), v_p(0), 0]}$$

MMS for a PIC simulation code

The modified model: $$\frac{\partial f_m}{\partial t} + v \cdot \frac{\partial f_m}{\partial x} + \frac{q}{m} E_m \cdot \frac{\partial f_m}{\partial v} = S_f$$

MMS for a PIC simulation code

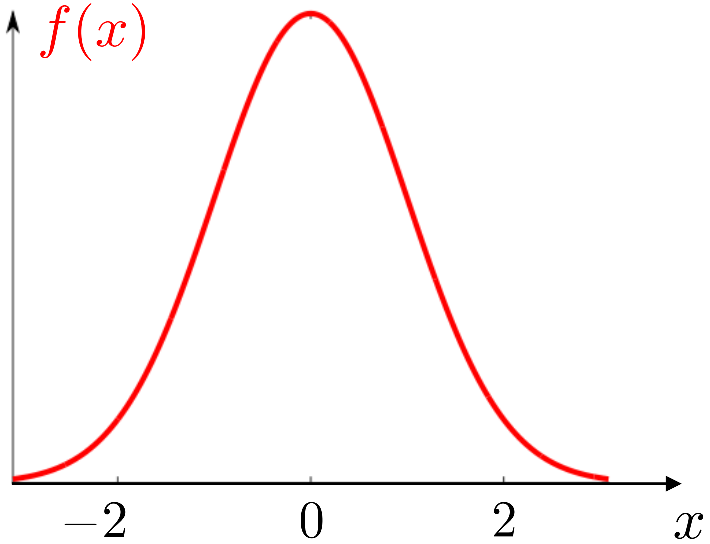
The modified model: $$\frac{\partial f_m}{\partial t} + v \cdot \frac{\partial f_m}{\partial x} + \frac{q}{m} E_m \cdot \frac{\partial f_m}{\partial v} = S_f$$

$$\frac{\partial E_m}{\partial x} = \frac{\rho}{\epsilon_0} + S_E$$

How to compare $f_m(x, v, t)$ with $f_N(x, v, t)$?

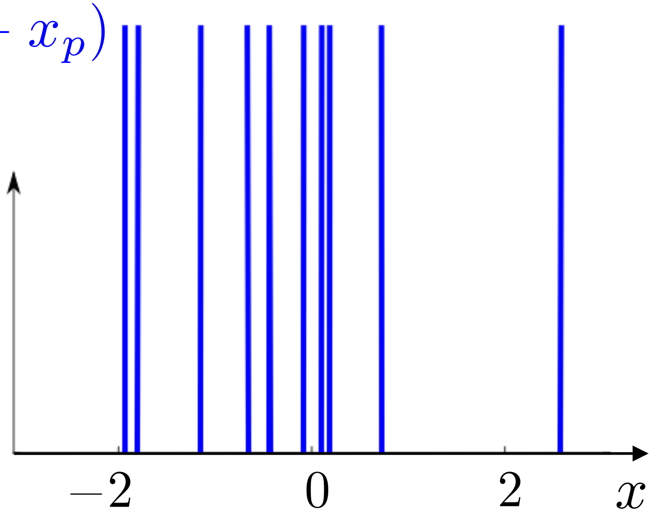
How to account for the statistical uncertainty?

1D example

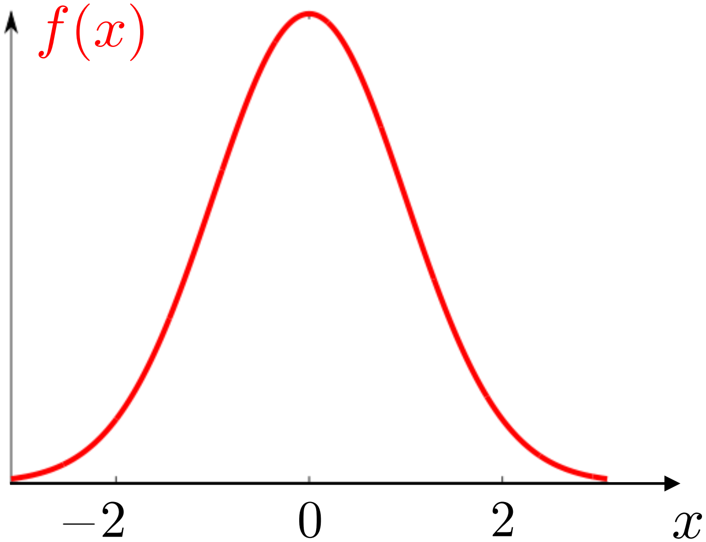


$$\Leftarrow ? \Rightarrow$$

$$f_N = \sum_{p=1}^{N} \frac{1}{N} \delta(x - x_p)$$

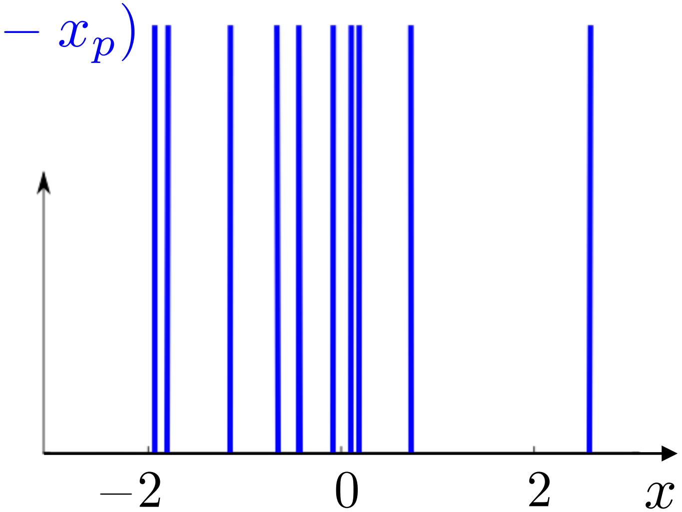


1D example

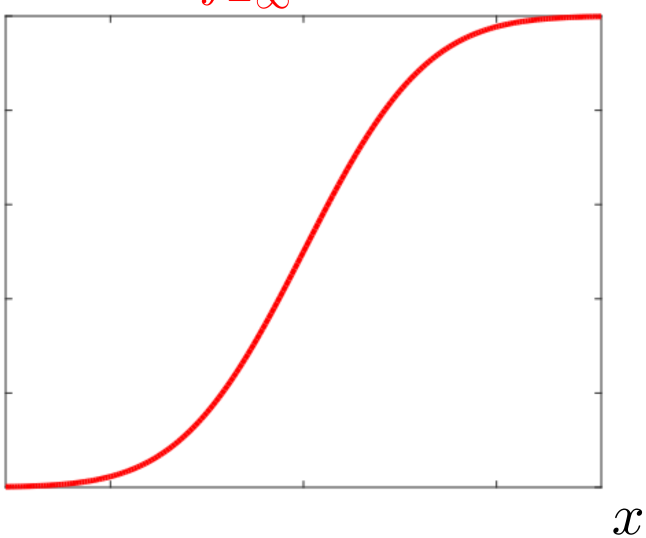


$$f_N = \sum_{p=1}^{N} \frac{1}{N}\delta(x - x_p)$$

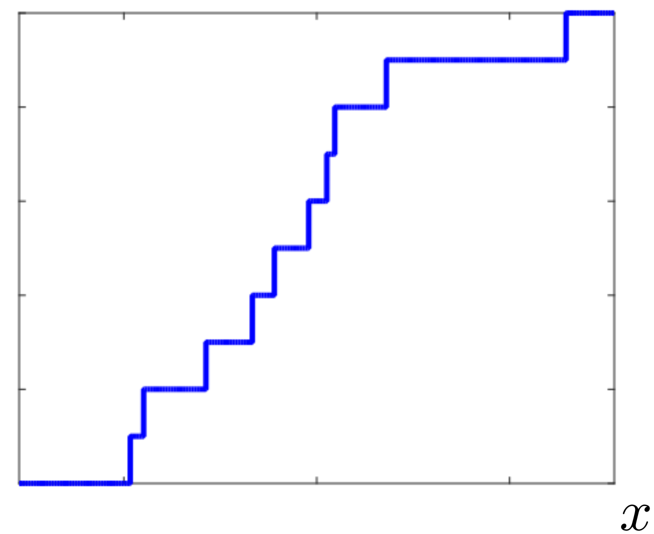
$$\Leftarrow ? \Rightarrow$$



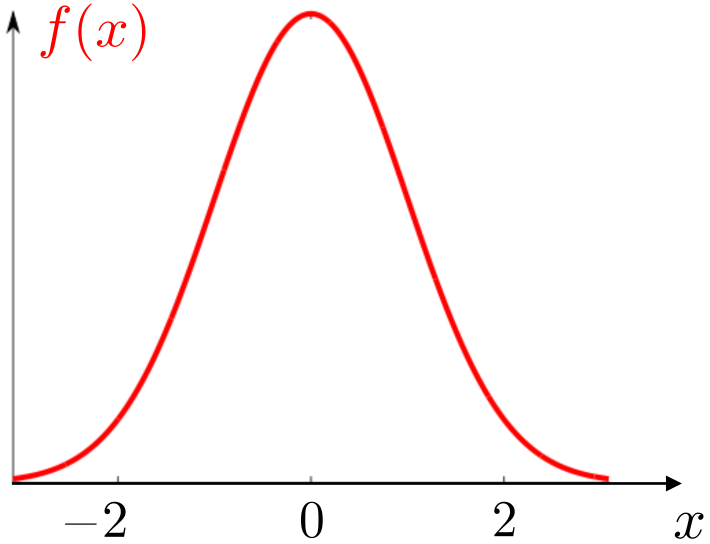
CDF = $\int_{-\infty}^{x} f(x')dx'$



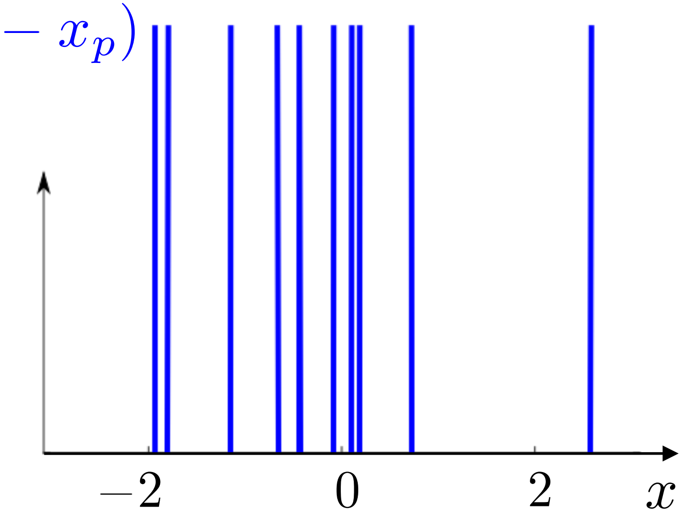
EDF



1D example

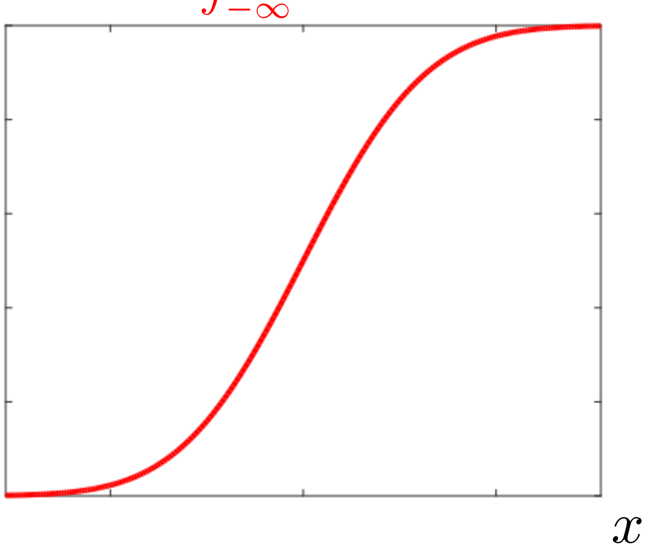


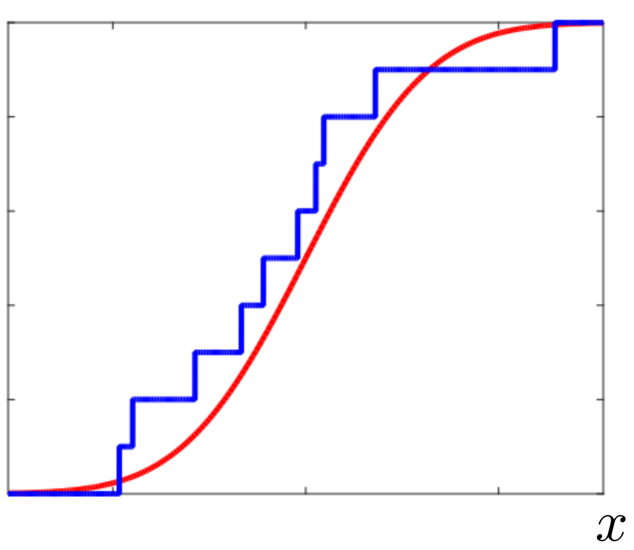
$$f_N = \sum_{p=1}^{N} \frac{1}{N}\delta(x - x_p)$$

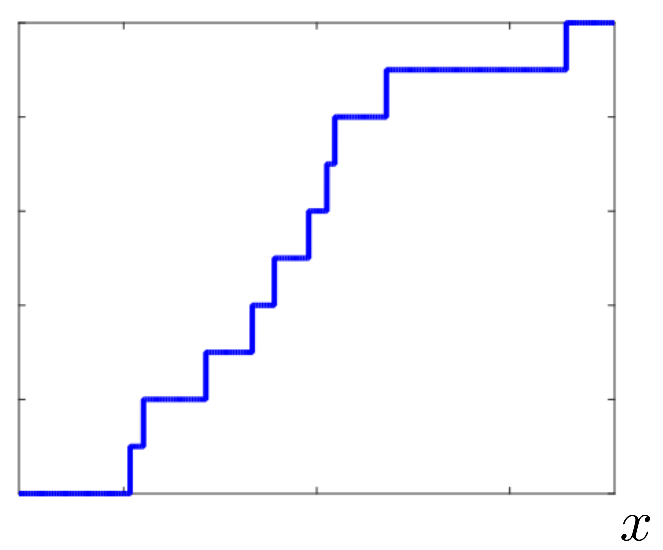


CDF = $\int_{-\infty}^{x} f(x')dx'$

EDF





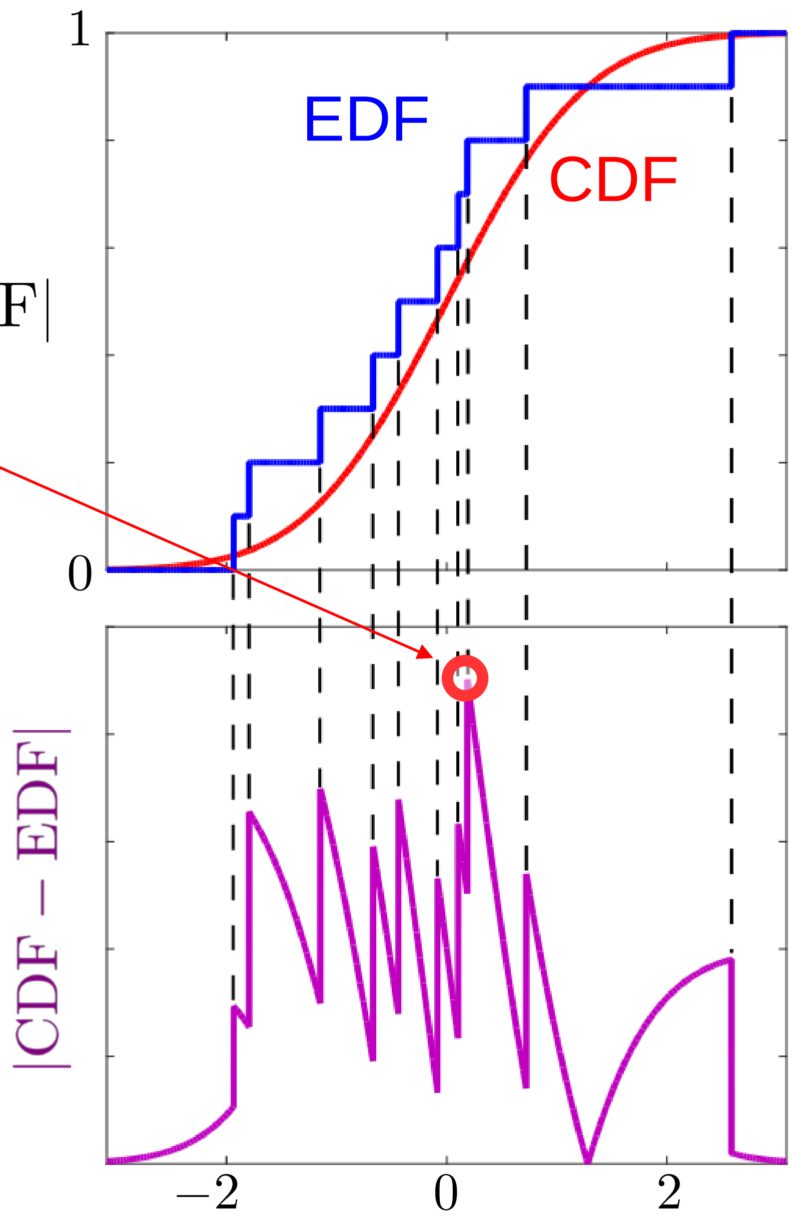


Kolmogorov–Smirnov statistic

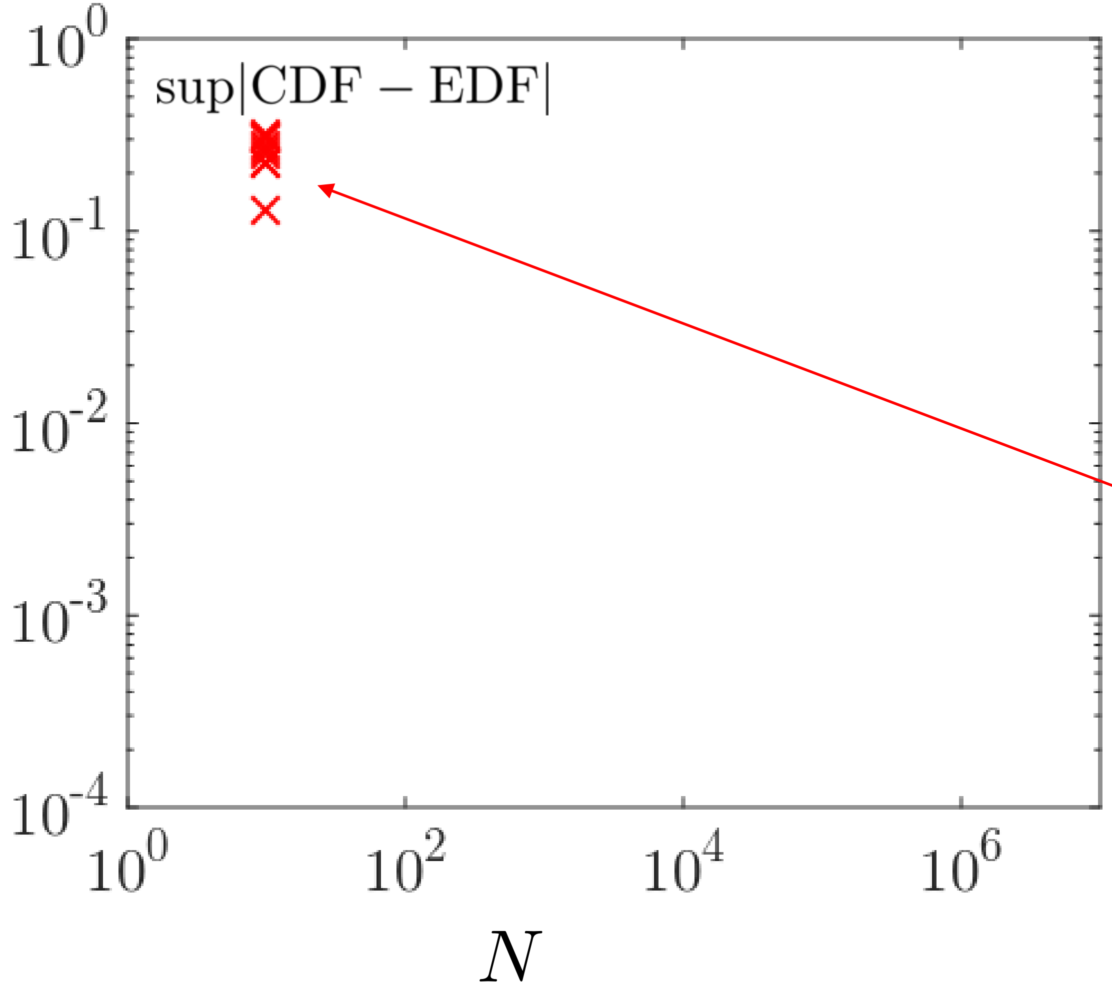
Distance between

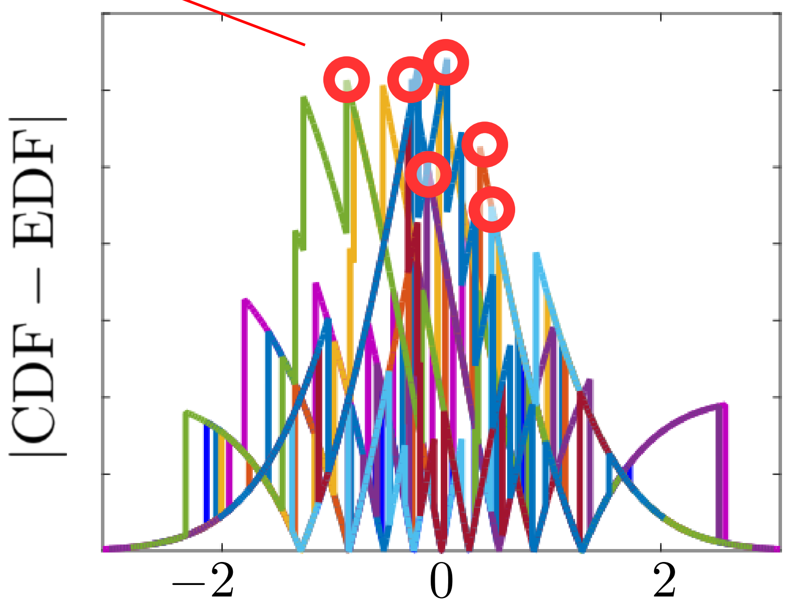
$f(x)$ and $f_N = \sum_{p=1}^{N} \frac{1}{N} \delta(x - x_p)$

defined as $d = \|f - f_N\| = \sup |\mathrm{CDF} - \mathrm{EDF}|$

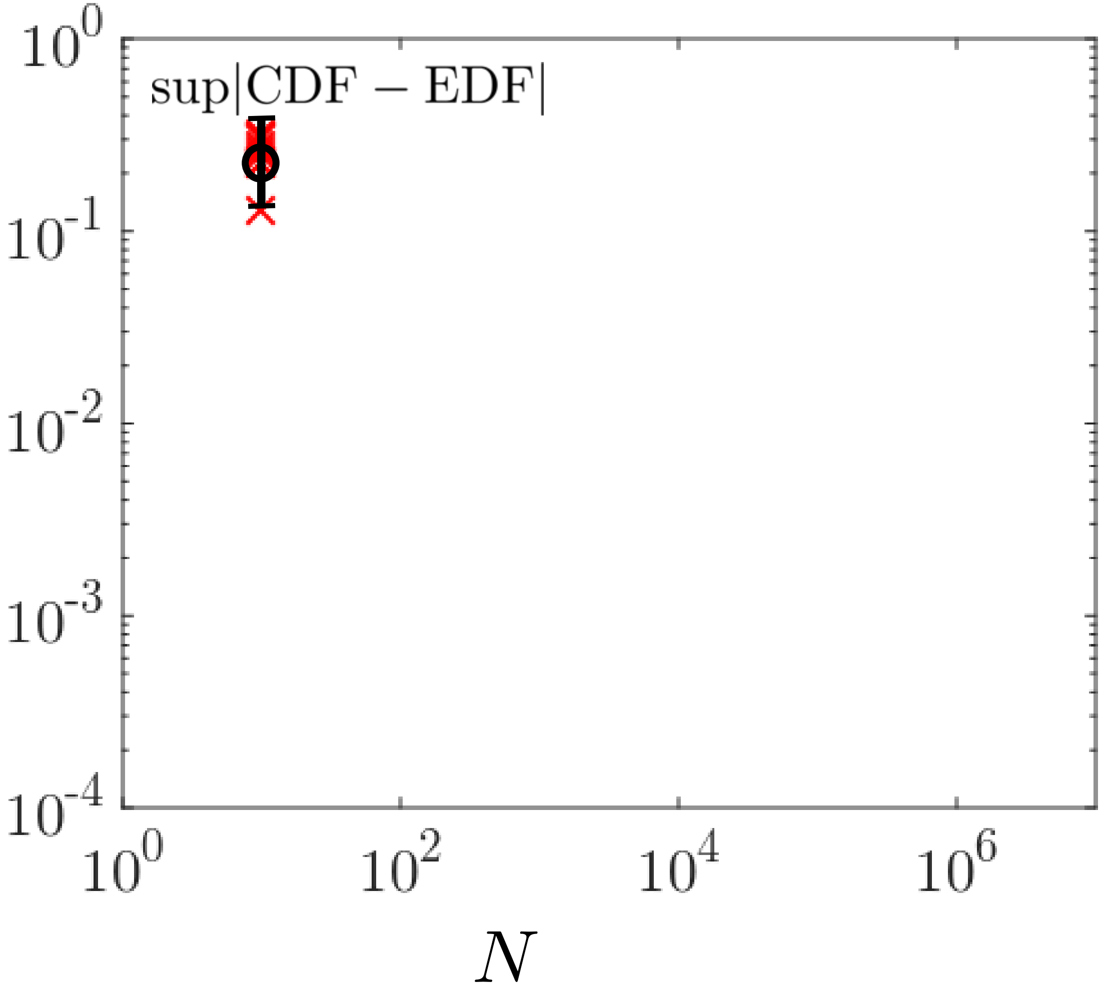


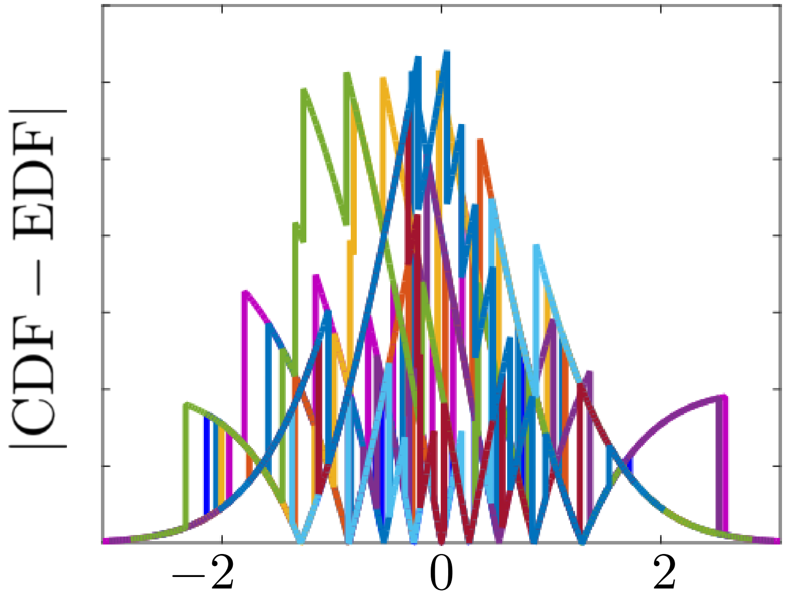
Kolmogorov–Smirnov statistic



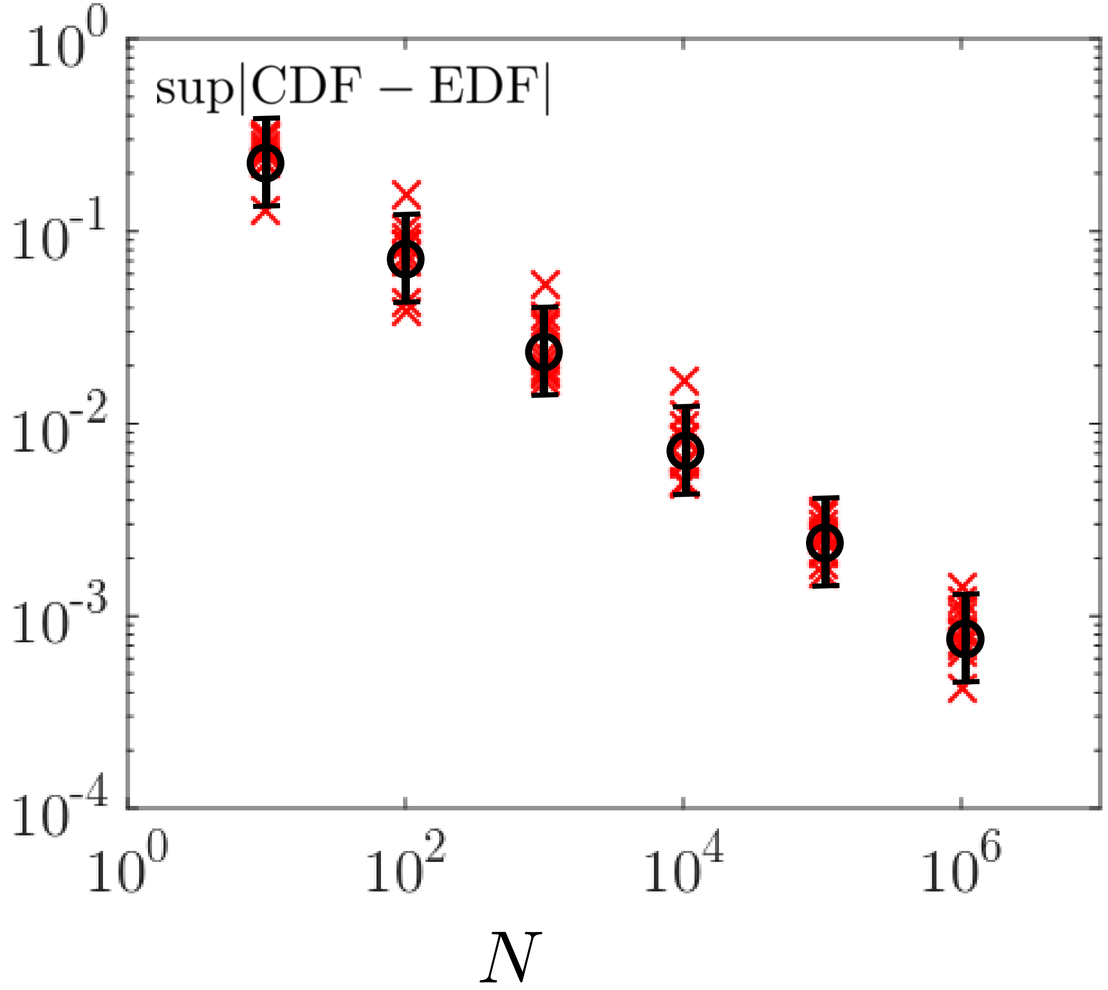


Kolmogorov–Smirnov statistic

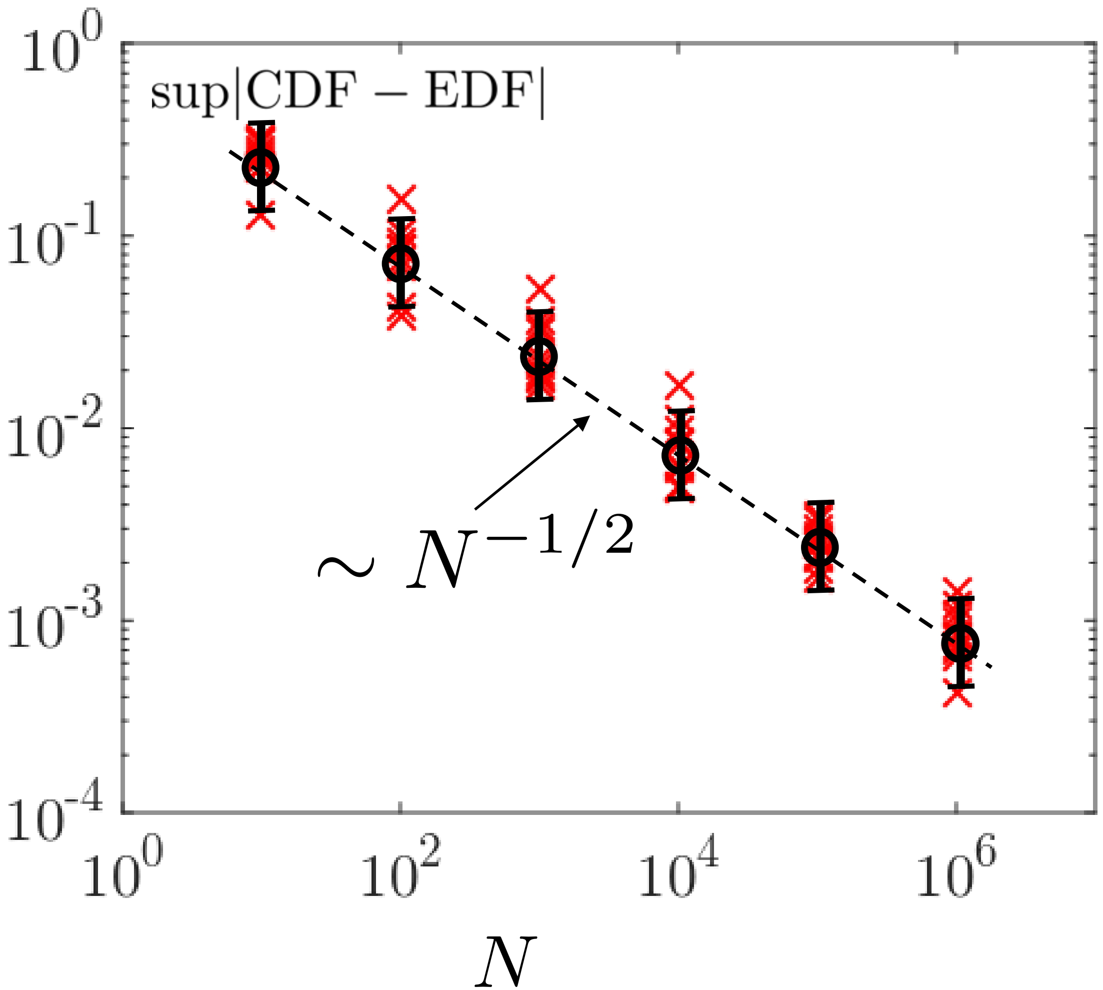




Kolmogorov–Smirnov statistic



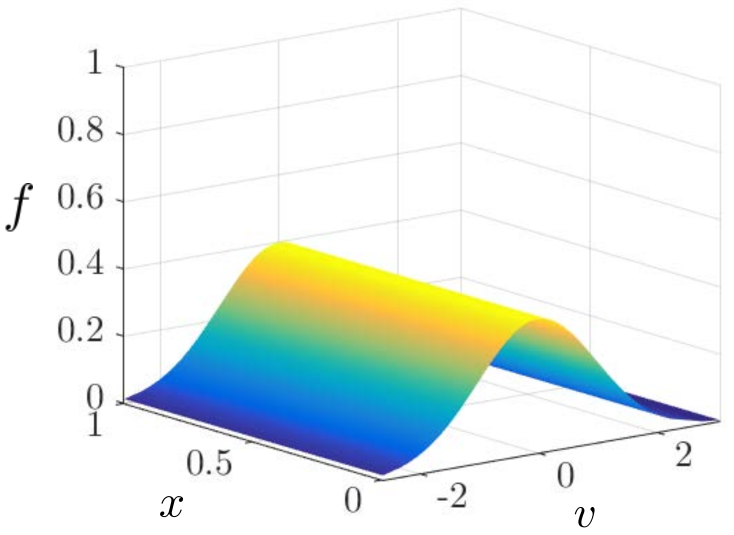
Kolmogorov–Smirnov statistic



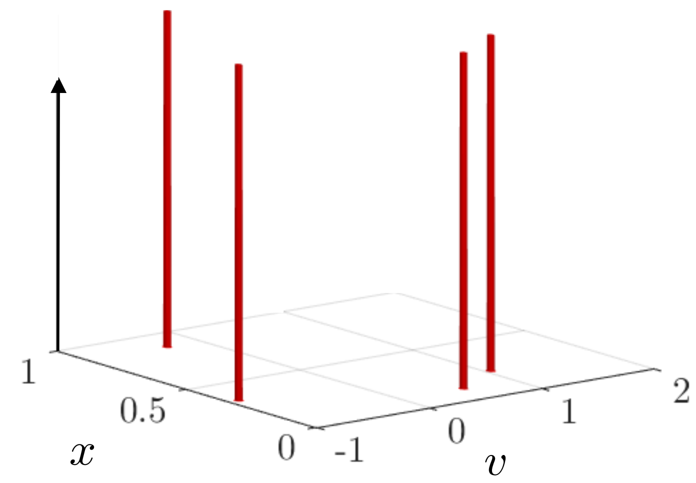
$$\sup|\mathrm{CDF} - \mathrm{EDF}| \propto N^{-1/2}$$

[Kolmogorov, *G.* Ist. Ital. Attuari. (1933); Smirnov, Ann. Math. Stat. (1948)]

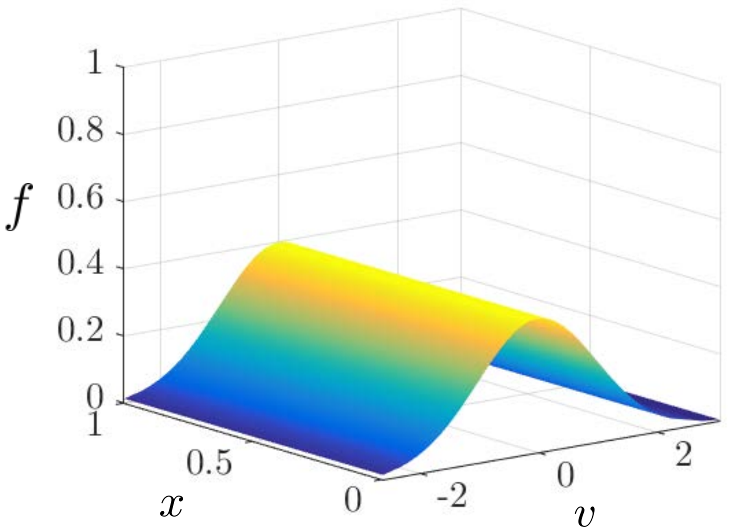
How to generalize to 2D case?



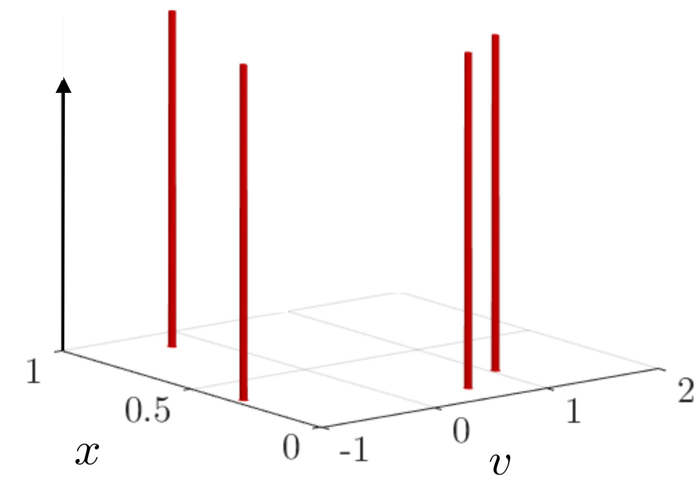
$\Leftarrow ? \Rightarrow$



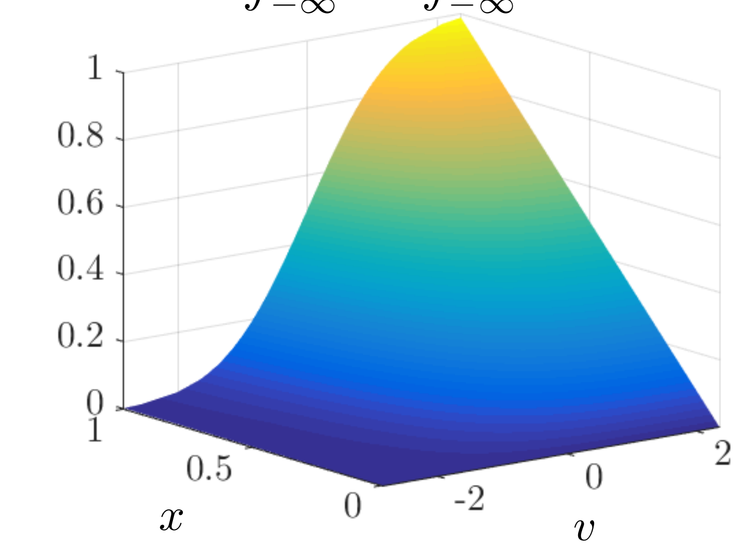
How to generalize to 2D case?



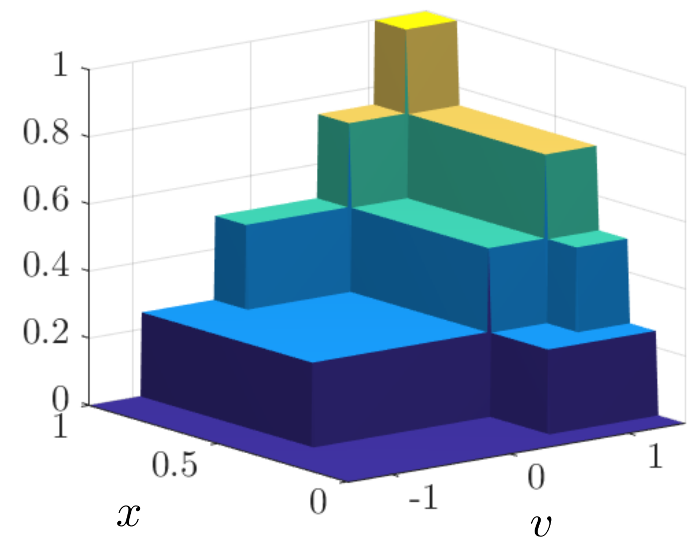
$\Leftarrow ? \Rightarrow$



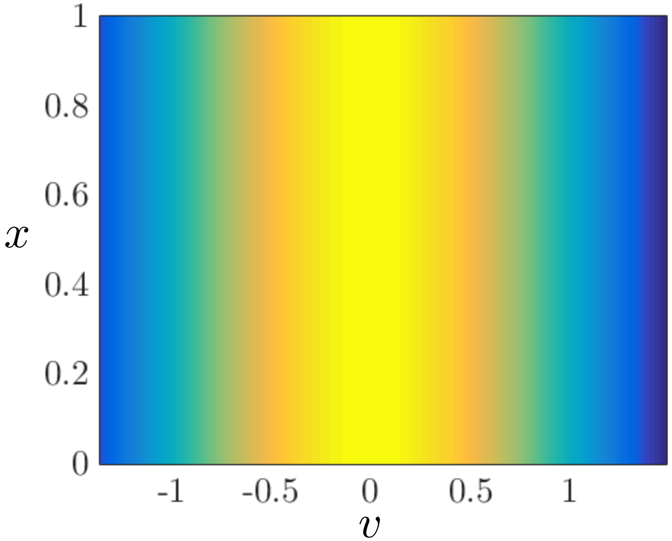
CDF $= \int_{-\infty}^{x} dx' \int_{-\infty}^{v} dv' f(x', v')$



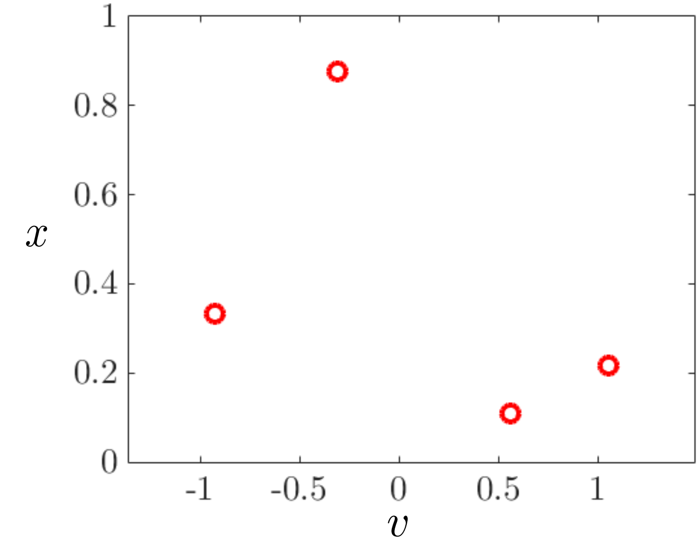
EDF



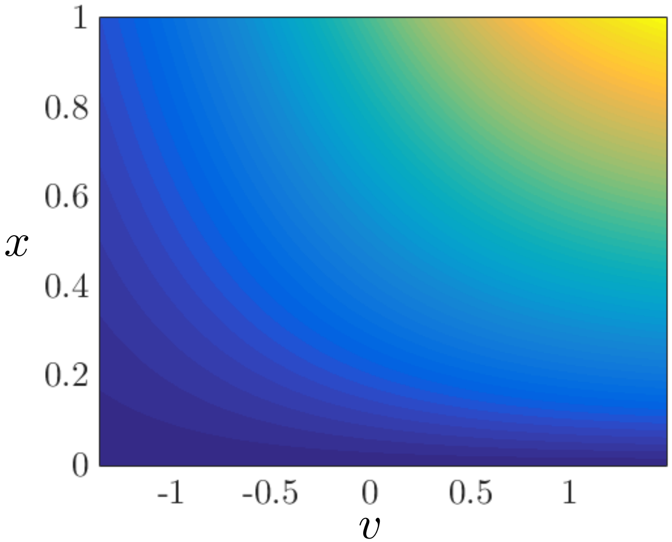
How to generalize to 2D case?



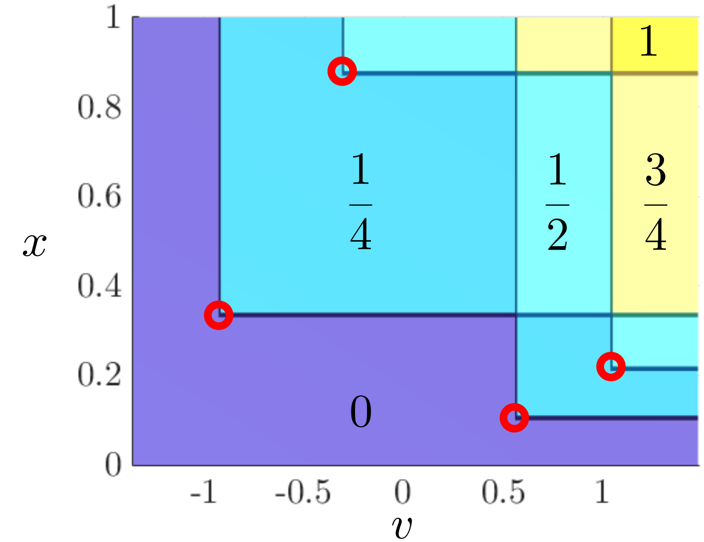
$\Leftarrow ? \Rightarrow$



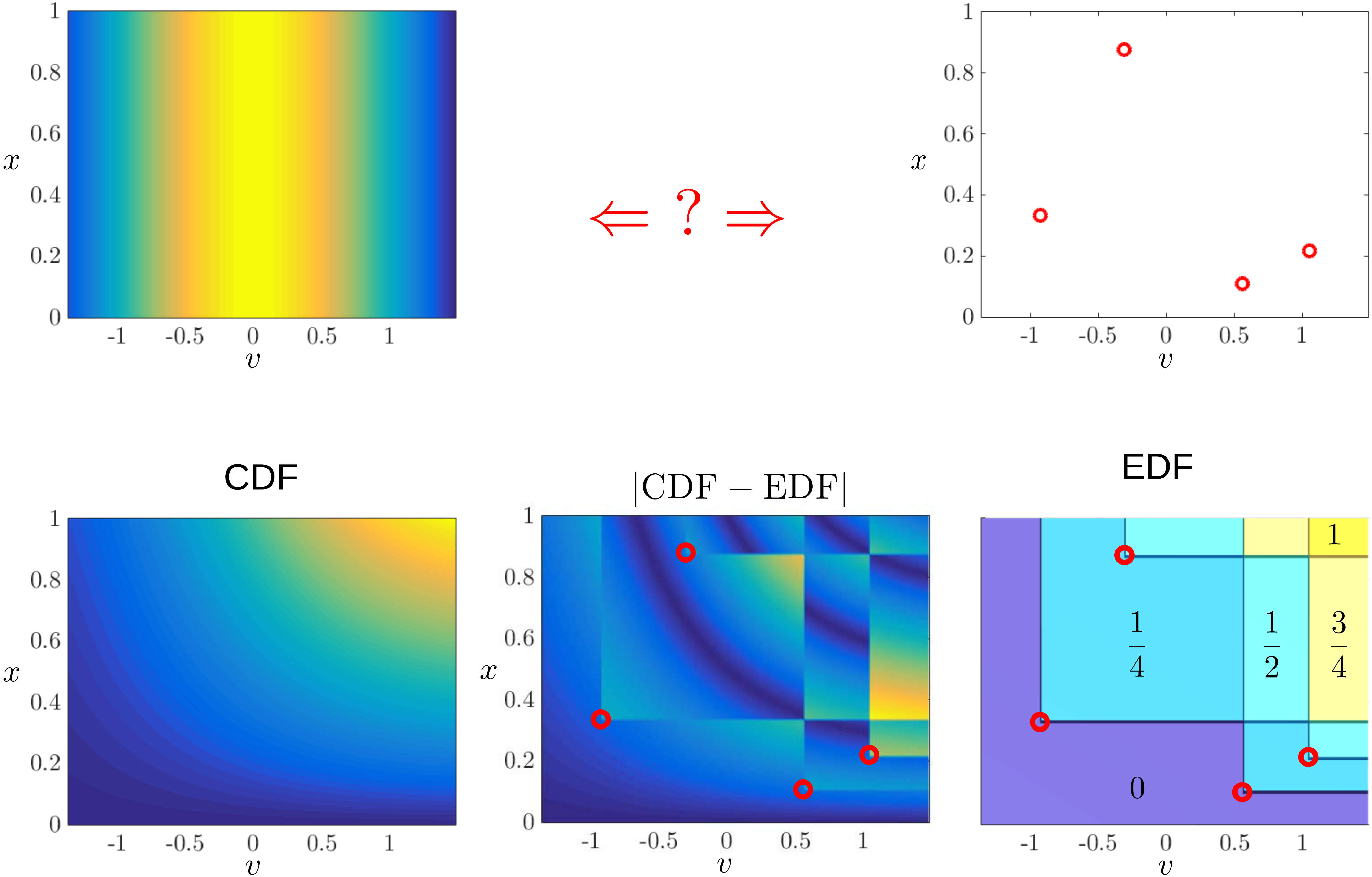
CDF



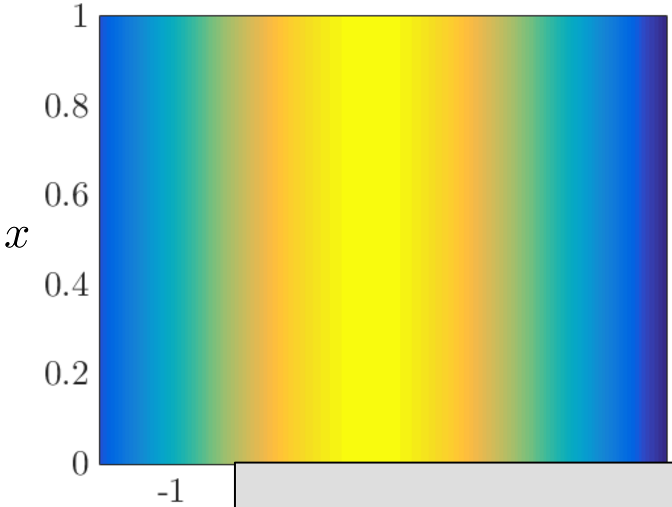
EDF



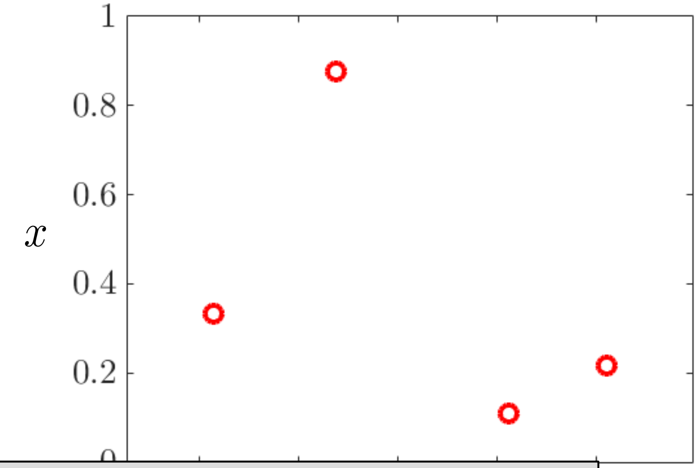
How to generalize to 2D case?



How to generalize to 2D case?

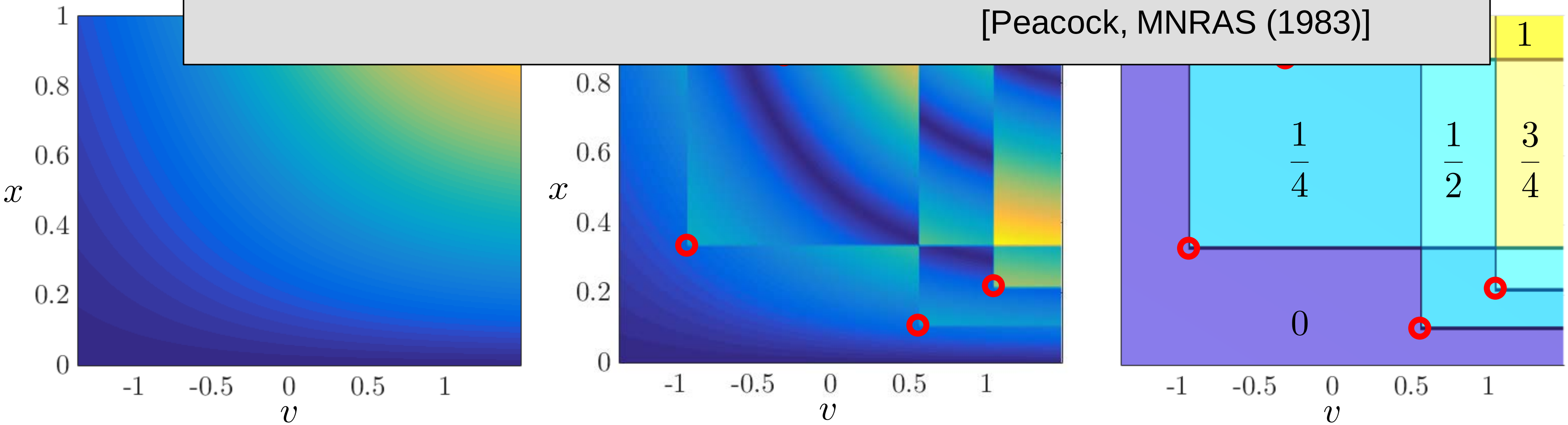


$\Leftarrow ? \Rightarrow$

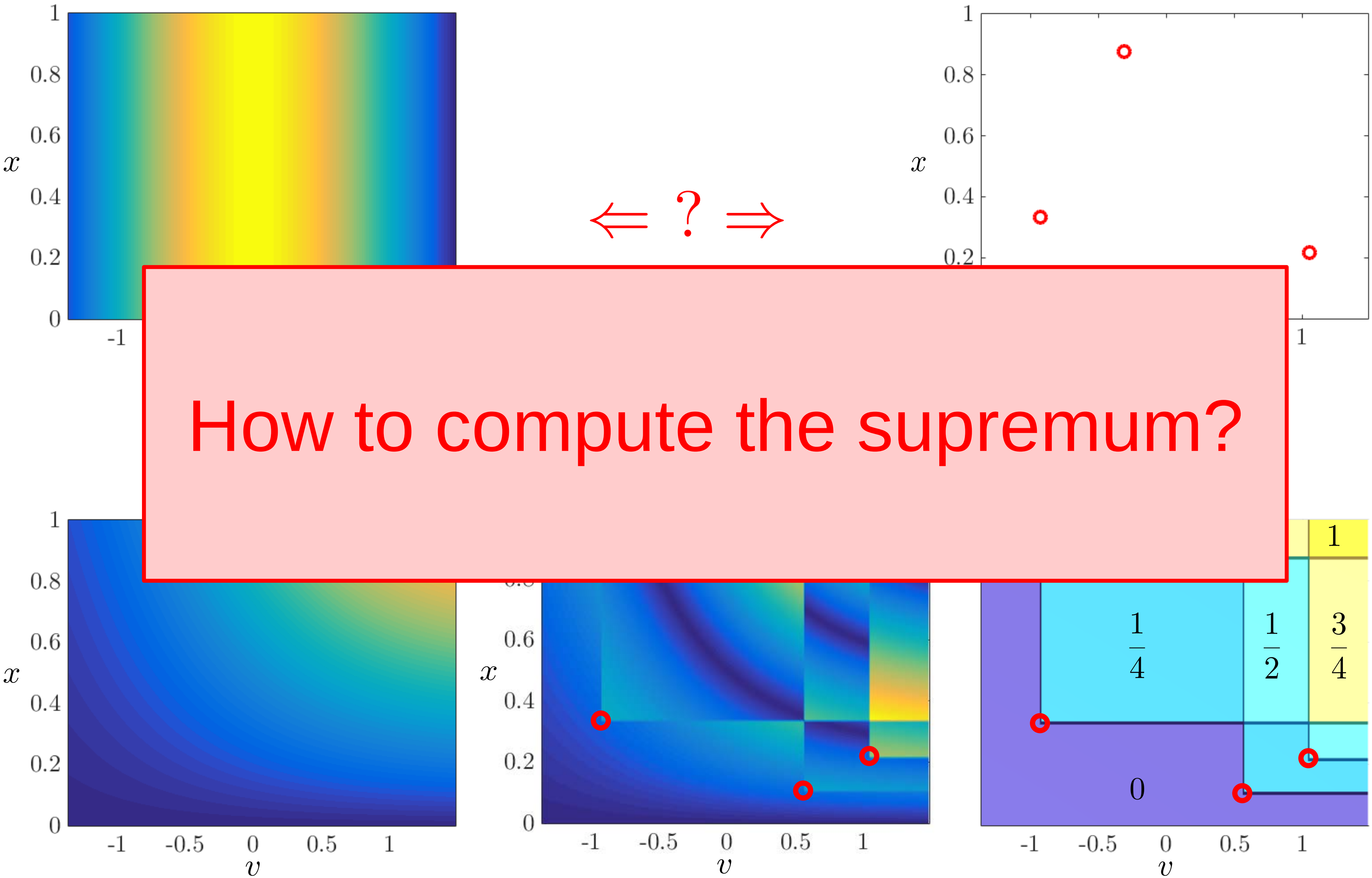


$$\sup |\mathrm{CDF} - \mathrm{EDF}| = \mathcal{O}(N^{-1/2})$$

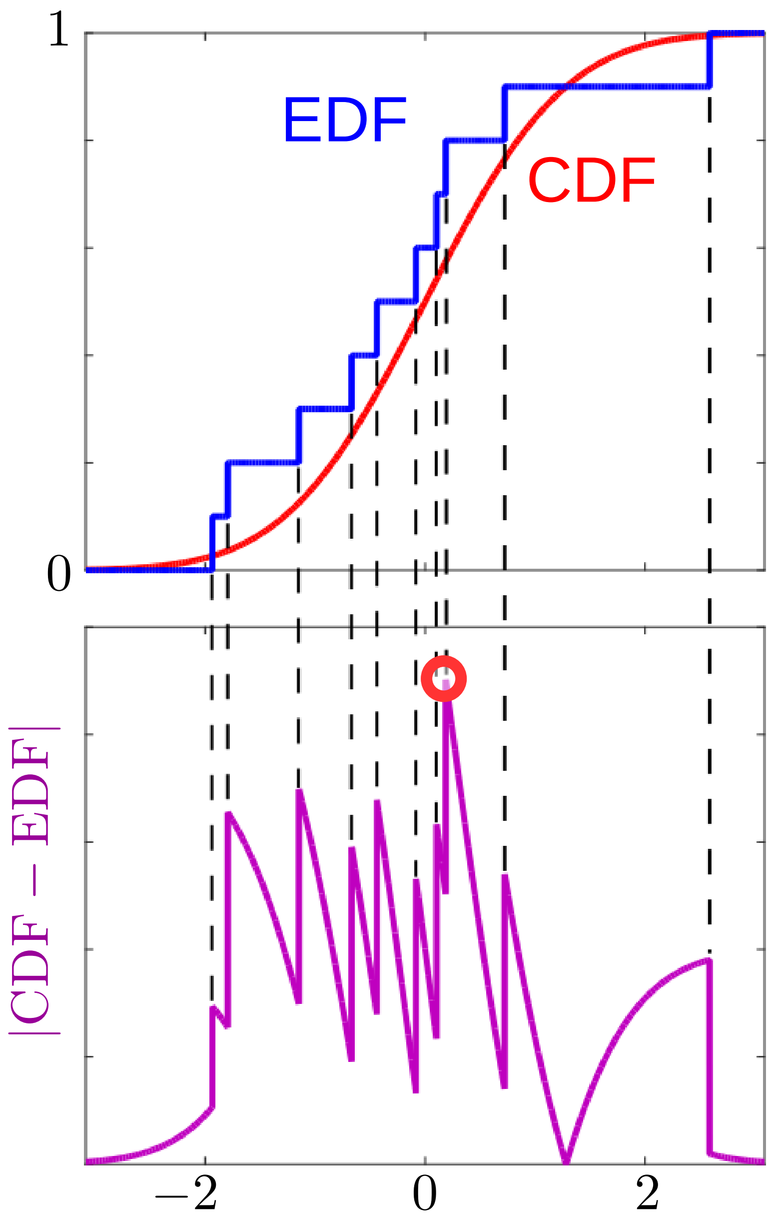
[Peacock, MNRAS (1983)]

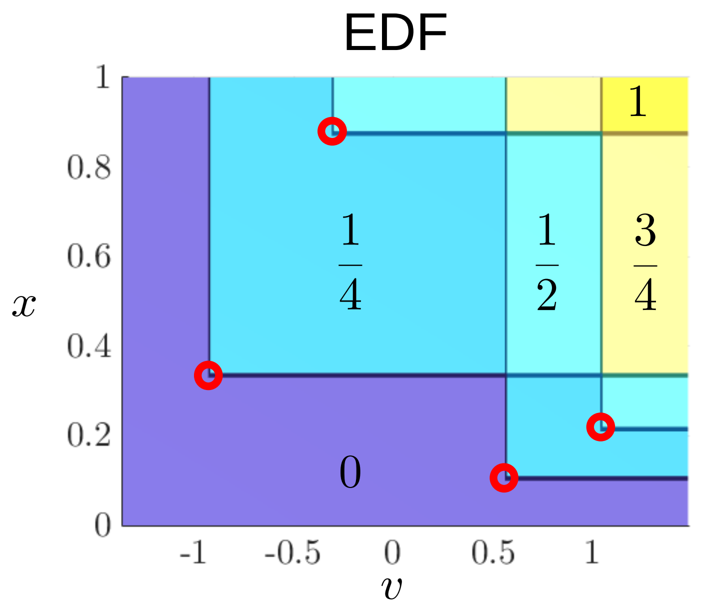


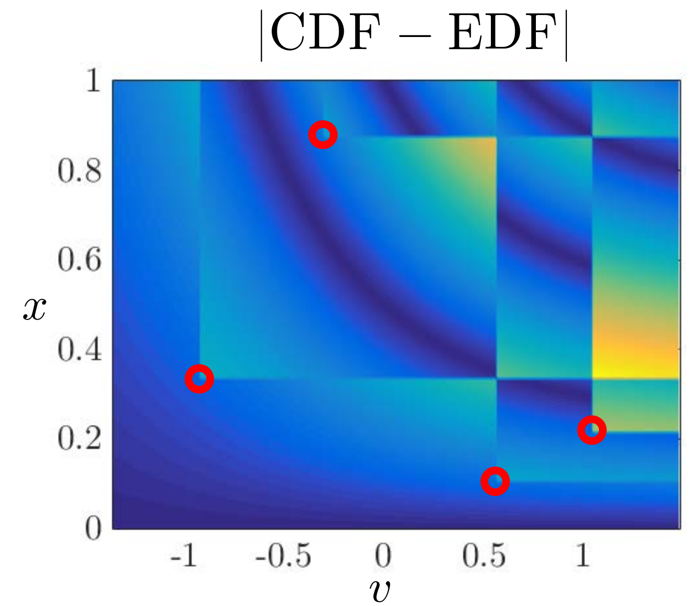
How to generalize to 2D case?



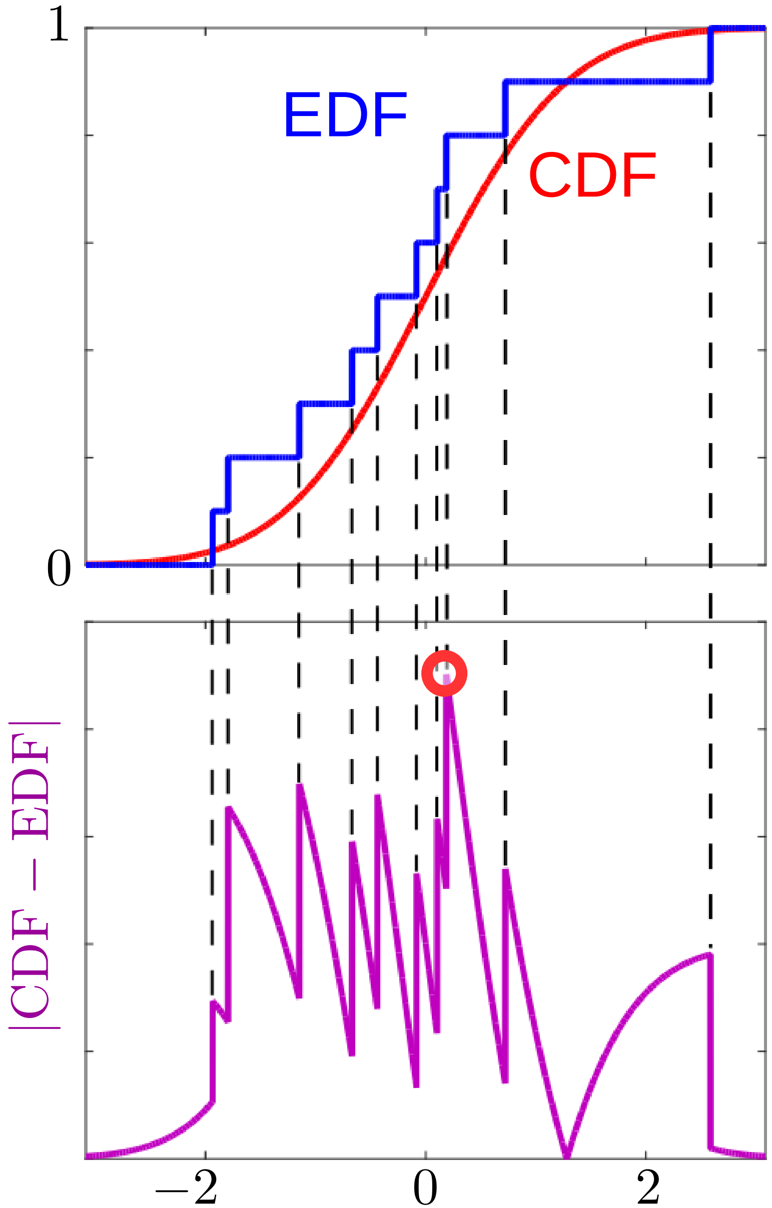
How to compute $\sup |\mathrm{CDF} - \mathrm{EDF}|$?

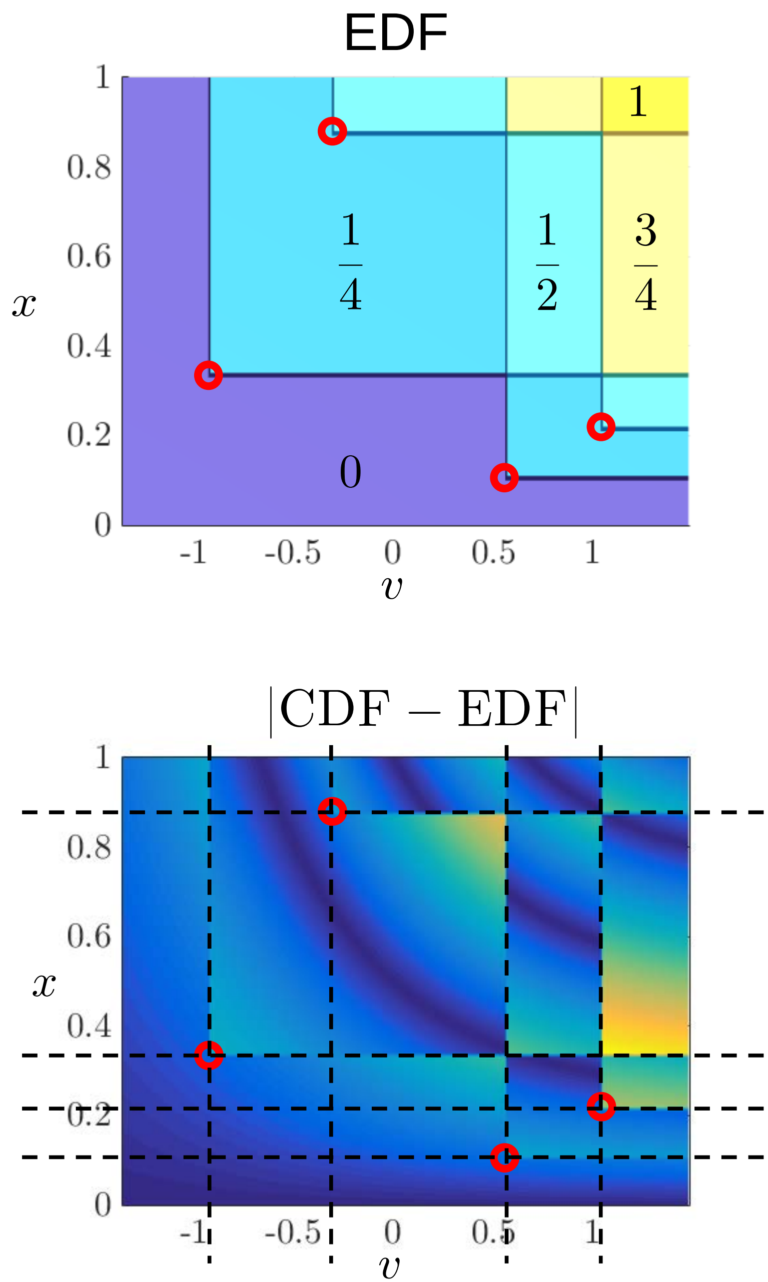




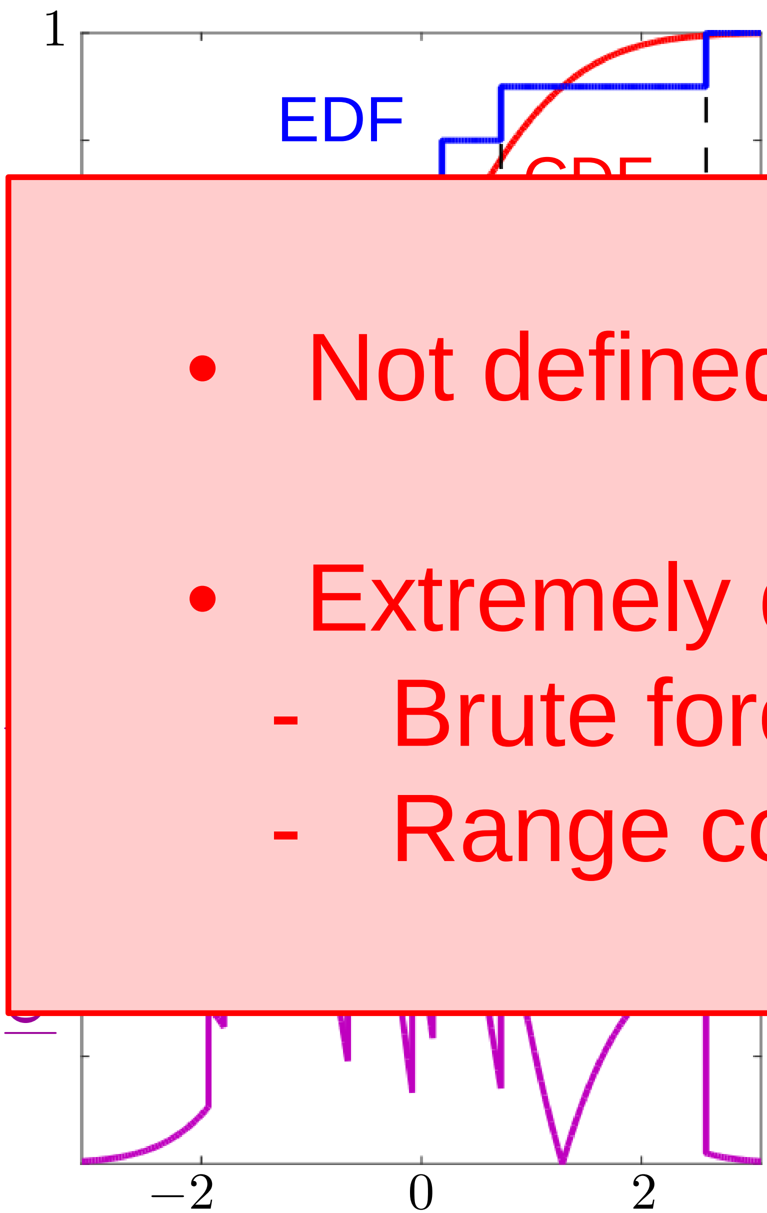


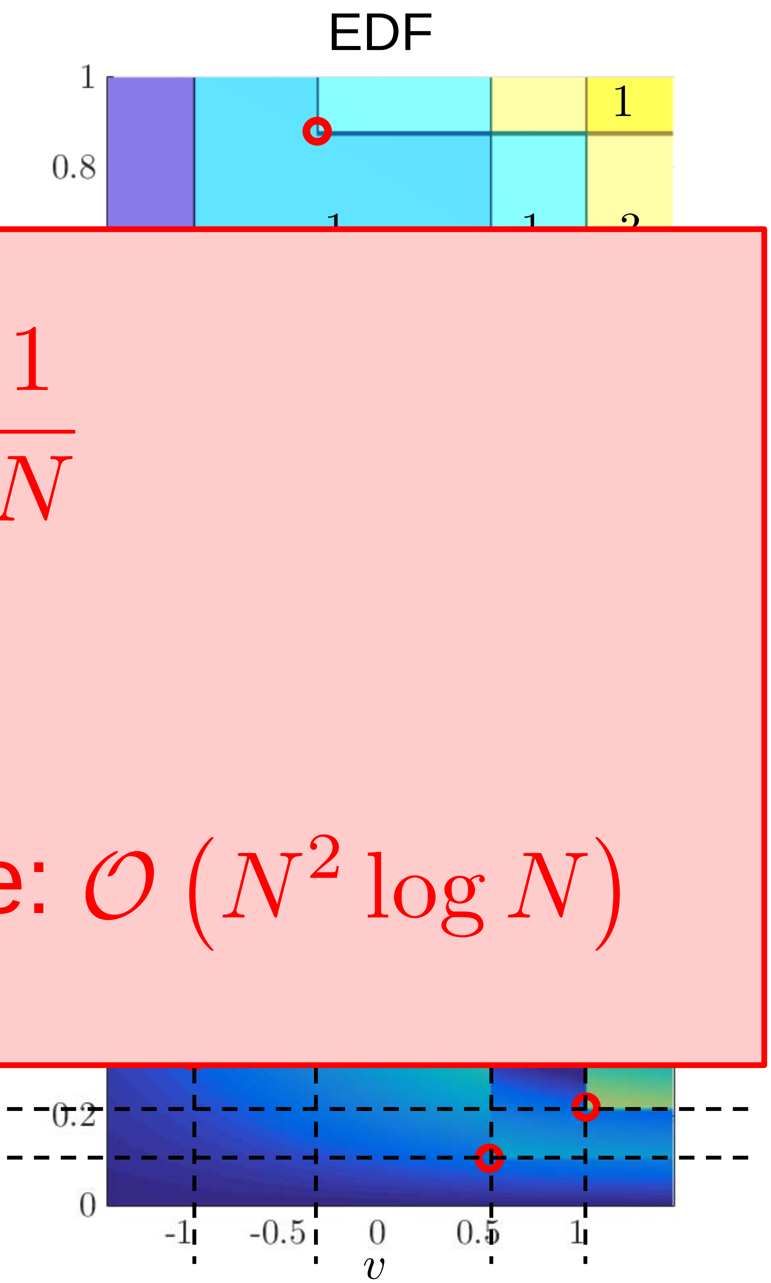
How to compute $\sup|\mathrm{CDF} - \mathrm{EDF}|$?





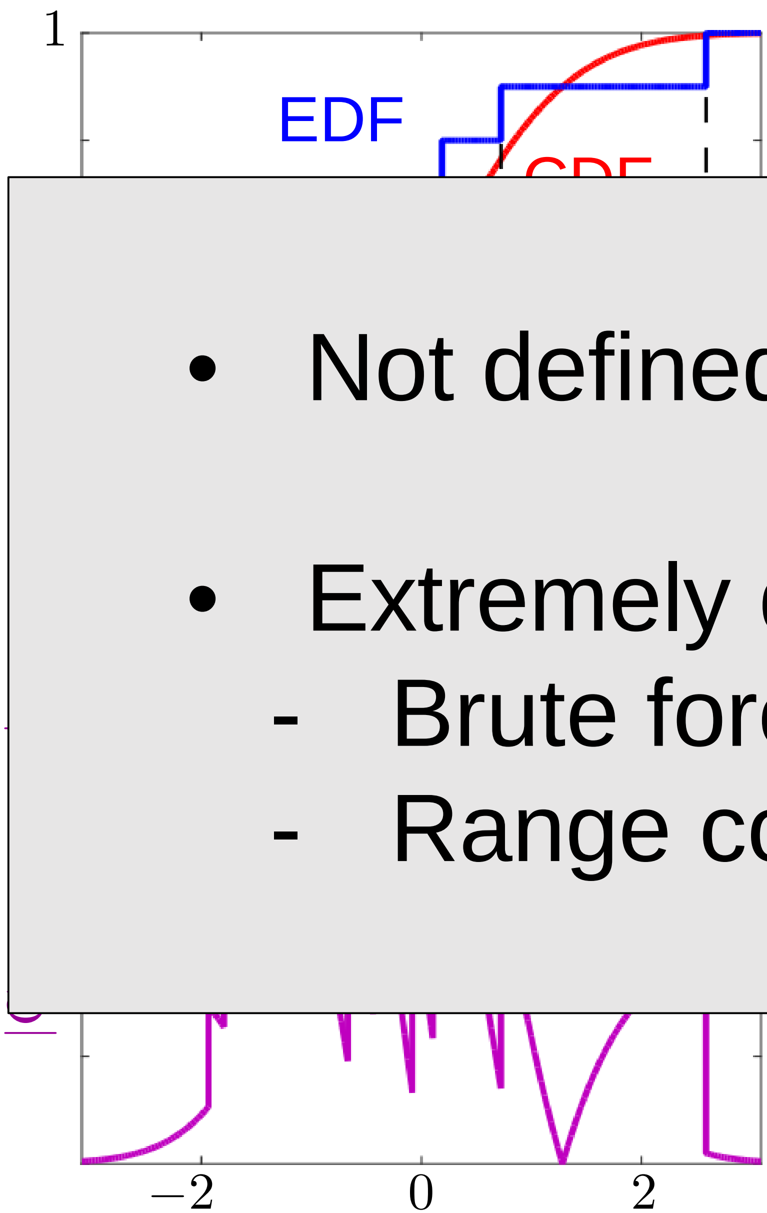
How to compute $\sup |\mathrm{CDF} - \mathrm{EDF}|$?

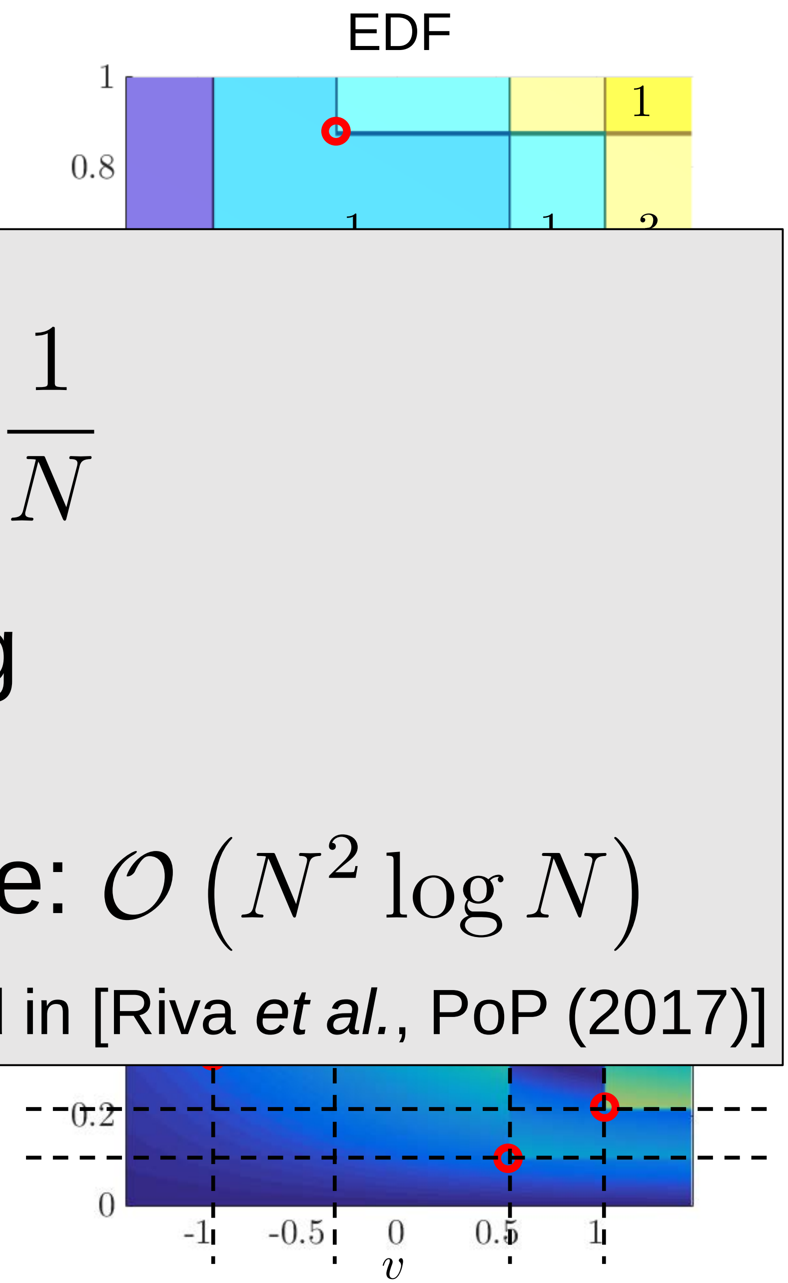




- Not defined for $w_p \neq \frac{1}{N}$
- Extremely demanding
  - Brute force: $\mathcal{O}(N^3)$
  - Range counting-tree: $\mathcal{O}\left(N^2 \log N\right)$

How to compute $\sup |\mathrm{CDF} - \mathrm{EDF}|$?





- Not defined for $w_p \neq \dfrac{1}{N}$

- Extremely demanding
  - Brute force: $\mathcal{O}(N^3)$
  - Range counting-tree: $\mathcal{O}\left(N^2 \log N\right)$

Addressed in [Riva *et al.*, PoP (2017)]

The PIC simulation code

Model equations:

$$\frac{\partial f_e}{\partial t} + v \cdot \frac{\partial f_e}{\partial x} - \frac{e}{m_e} E \cdot \frac{\partial f_e}{\partial v} = 0$$

$$\frac{\partial E}{\partial x} = \frac{\rho}{\epsilon_0}$$

- Interpolation scheme (Δx): first-order weighting (CIC PIC)
- Poisson solver (Δx): second order centered finite differences
- Time integration (Δt): Leapfrog integration scheme

The PIC simulation code

Model equations:

$$\frac{\partial f_e}{\partial t} + v \cdot \frac{\partial f_e}{\partial x} - \frac{e}{m_e} E \cdot \frac{\partial f_e}{\partial v} = 0$$

$$\frac{\partial E}{\partial x} = \frac{\rho}{\epsilon_0}$$

- Interpolation scheme (Δx): first-order weighting (CIC PIC)
- Poisson solver (Δx): second order centered finite differences
- Time integration (Δt): Leapfrog integration scheme

$$\Rightarrow \epsilon_h = \mathcal{O}(\Delta x^2) + \mathcal{O}(\Delta t^2) + \mathcal{O}(N^{-1/2})$$

Results: PIC code verification

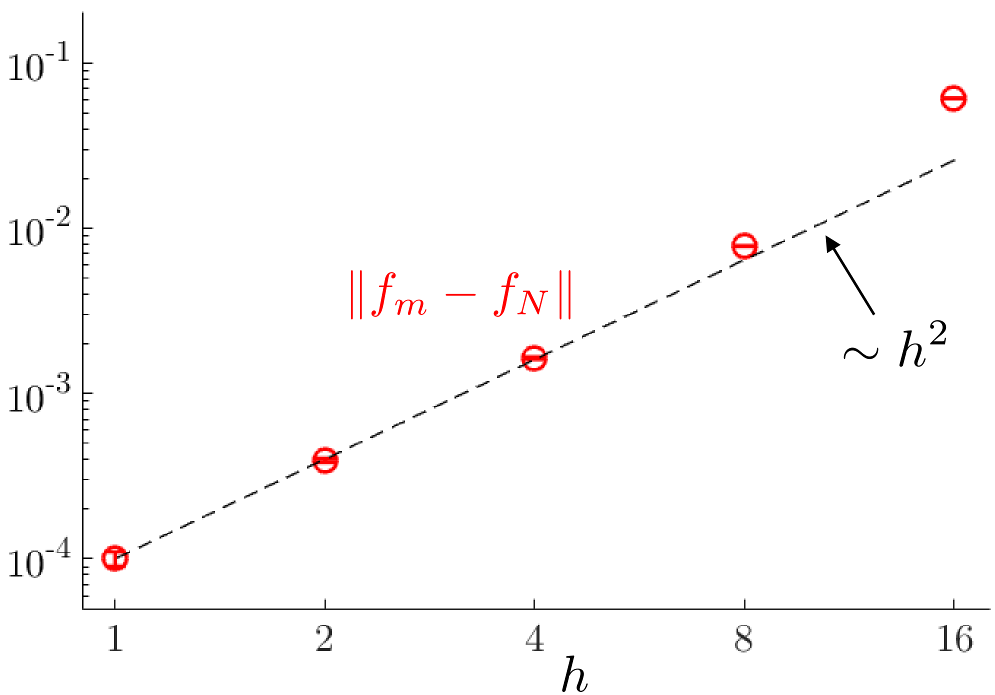
- Choose $f_m,\ E_m$
- Compute $S_f,\ S_E$
- Choose $\Delta x_0,\ \Delta t_0,\ N_0$
- Define $h = \dfrac{\Delta x}{\Delta x_0} = \dfrac{\Delta t}{\Delta t_0} = \left(\dfrac{N}{N_0}\right)^{-1/4}$
- Compute $\epsilon_h = \|f_m - f_N\|$ for different h
- Verify $\epsilon_h = Ch^2 + O(h^3)$

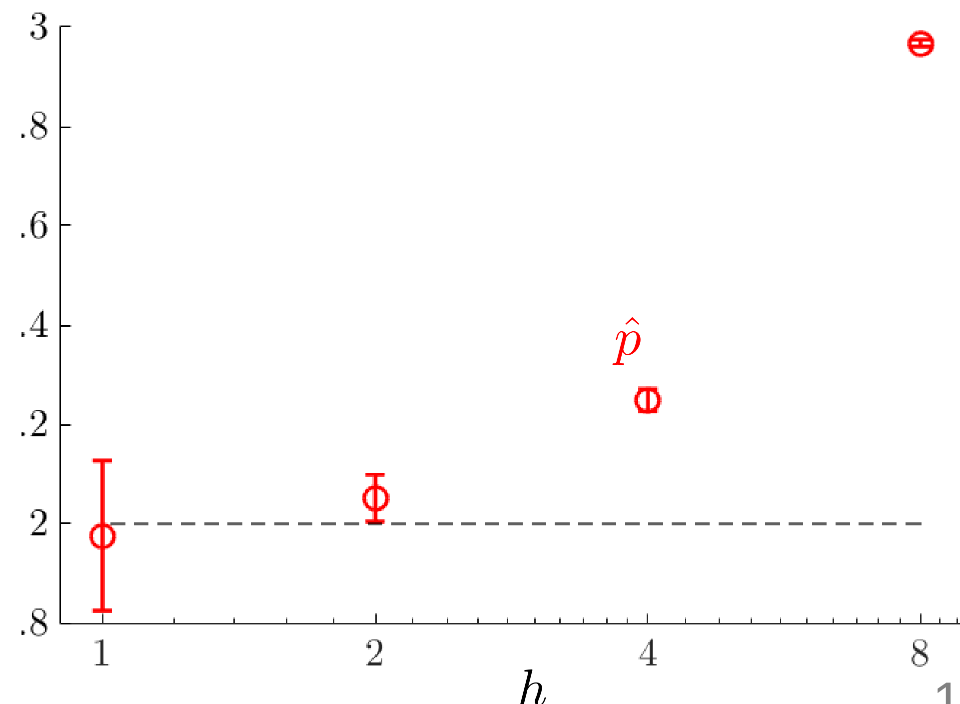
Notice: ϵ_h is affected by statistical uncertainty

$\Longrightarrow$ Repeat simulations with different random number generator seeds

Results: PIC code verification

- Choose $f_m,\ E_m$
- Compute $S_f,\ S_E$
- Choose $\Delta x_0,\ \Delta t_0,\ N_0$
- Define $h = \dfrac{\Delta x}{\Delta x_0} = \dfrac{\Delta t}{\Delta t_0} = \left(\dfrac{N}{N_0}\right)^{-1/4}$
- Compute $\epsilon_h = \|f_m - f_N\|$ for different h
- Verify $\epsilon_h = Ch^2 + O(h^3)$





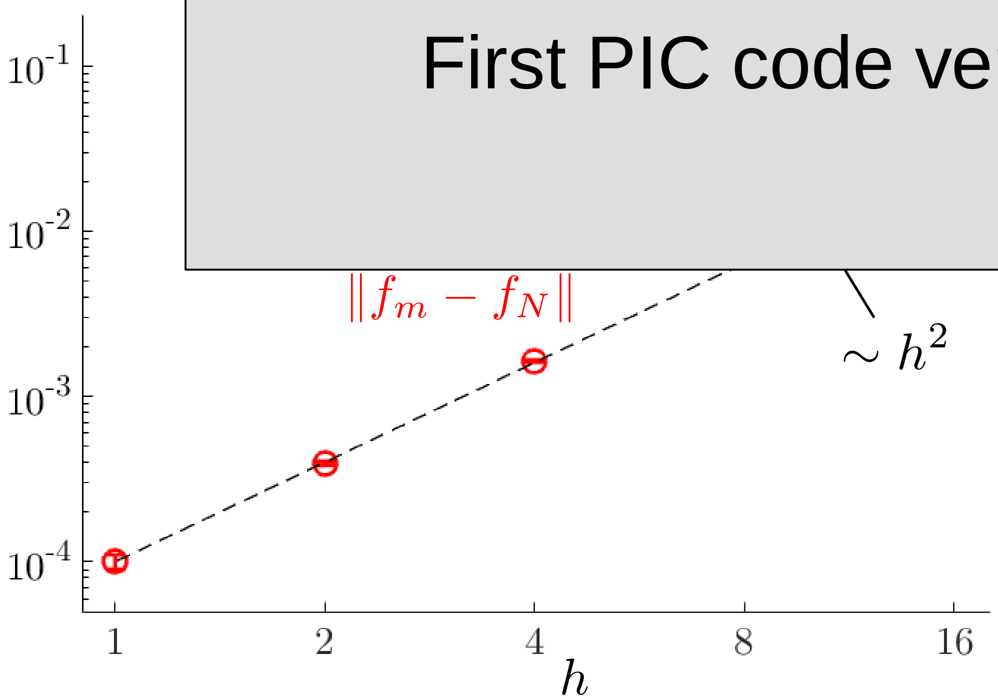
Results: PIC code verification

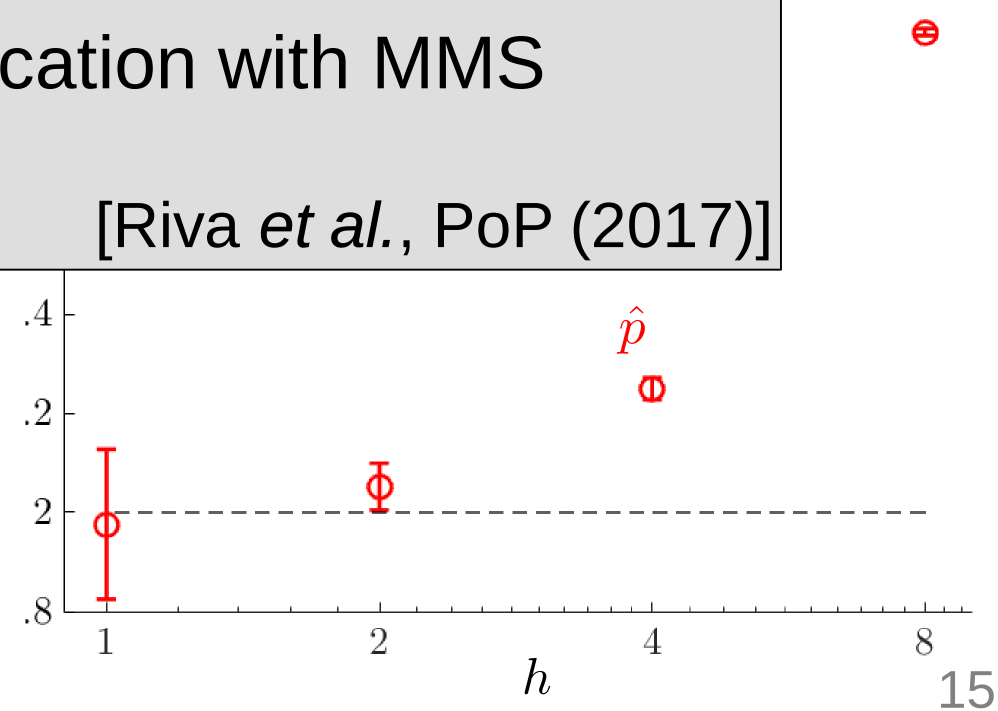
- Choose $f_m,\ E_m$
- Compute $S_f,\ S_E$
- Choose $\Delta x_0,\ \Delta t_0,\ N_0$
- Defin $\quad \Delta x \qquad \Delta t \qquad \left(N \right)^{-1/4}$
- Comp
- Verify

The PIC simulation code is verified!

First PIC code verification with MMS

[Riva *et al.*, PoP (2017)]





Verification Procedures

How to rigorously ensure that a simulation code is bug-free?

Code Verification

How to estimate the numerical uncertainty affecting simulation results?

Solution Verification

Sources of numerical uncertainties

1. Round-off
   → *Finite number of digits*

2. Iterative schemes
   → *Termination with finite residual*

3. Finite statistics
   → *E.g. a finite number of markers in representing a distribution function*

4. Discretization
   → *Grids with finite resolution*

5. Post-processing tools
   → *Evaluating observables from simulation results*

Sources of numerical uncertainties

1. Round-off
   → *Finite number of digits*

2. Iterative schemes
   → *Termination with finite residual*

3. Finite statistics
   → *E.g. a finite number of markers in representing a distribution function*

4. Discretization
   → *Grids with finite resolution*

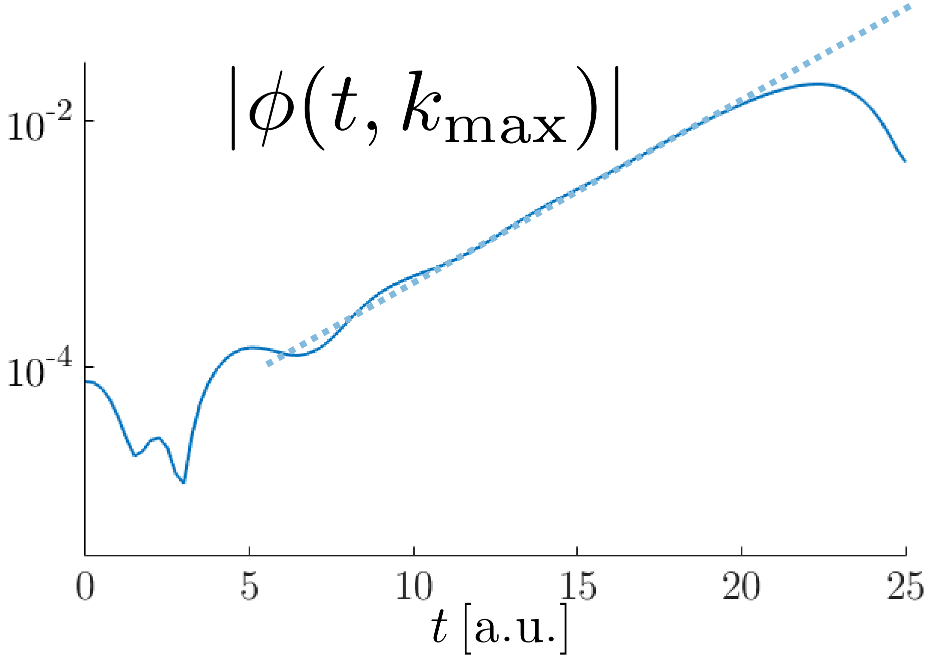
5. Post-processing tools
   → *Evaluating observables from simulation results*

Two-stream instability

Linear growth rate γ and its uncertainty?

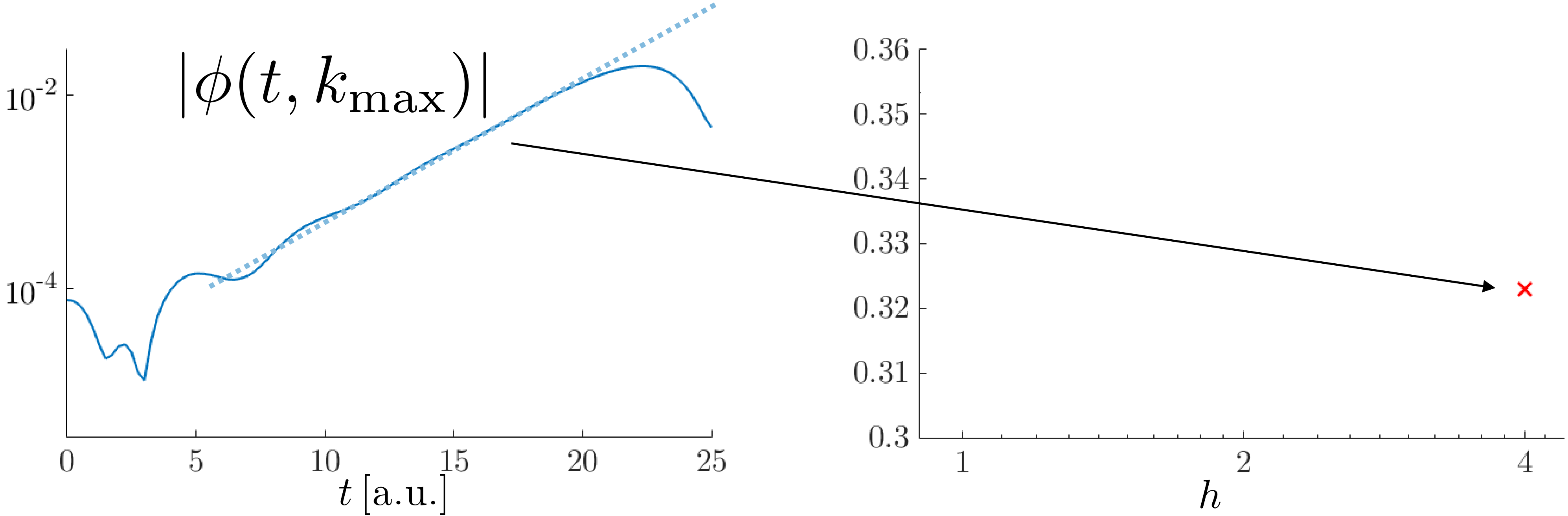
Post-processing uncertainty

- Choose $\Delta x,\ \Delta t,\ N$ and perform a simulation
- Get γ



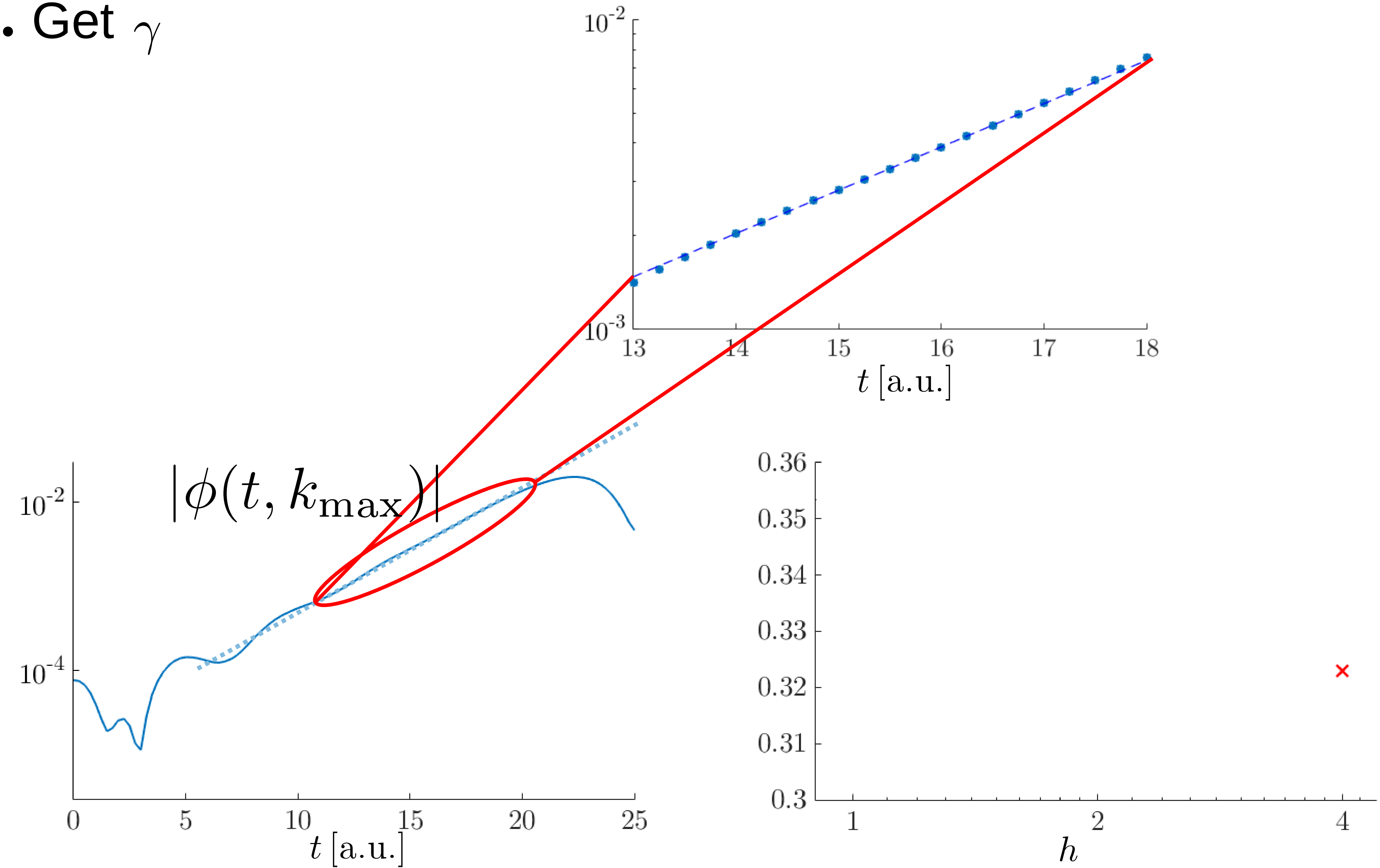
Post-processing uncertainty

- Choose Δx, Δt, N and perform a simulation
- Get γ



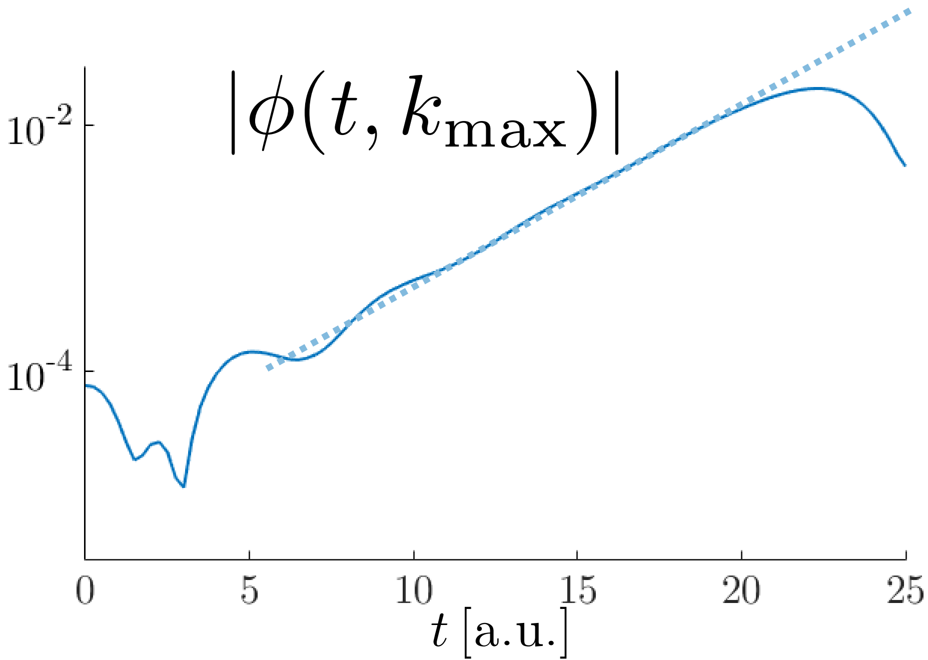
Post-processing uncertainty

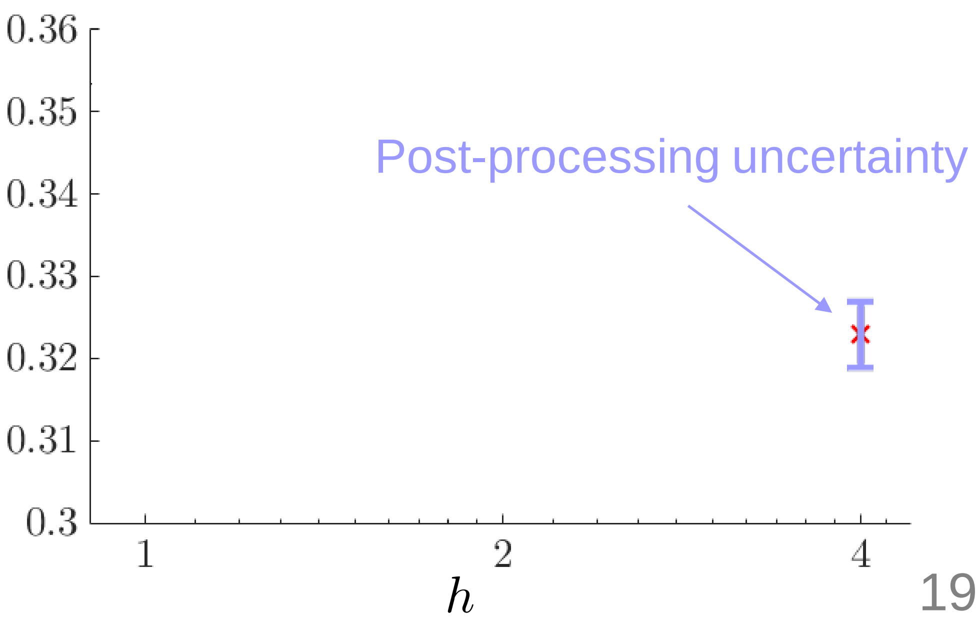
- Choose Δx, Δt, N and perform a simulation
- Get γ



Post-processing uncertainty

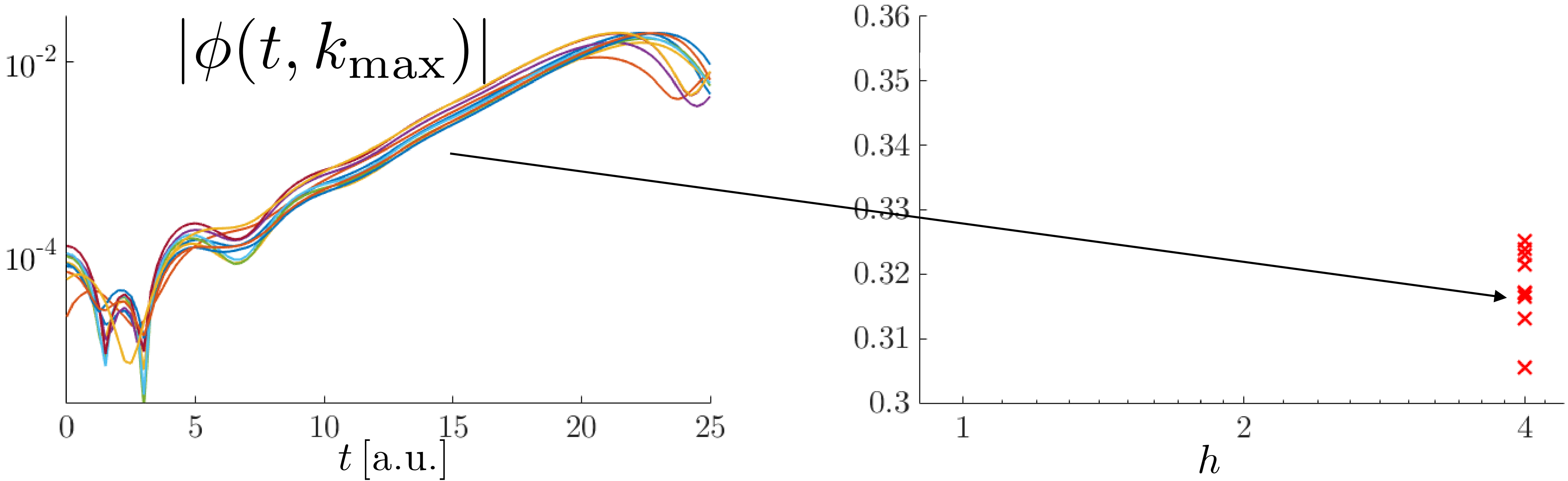
- Choose $\Delta x,\ \Delta t,\ N$ and perform a simulation
- Get γ





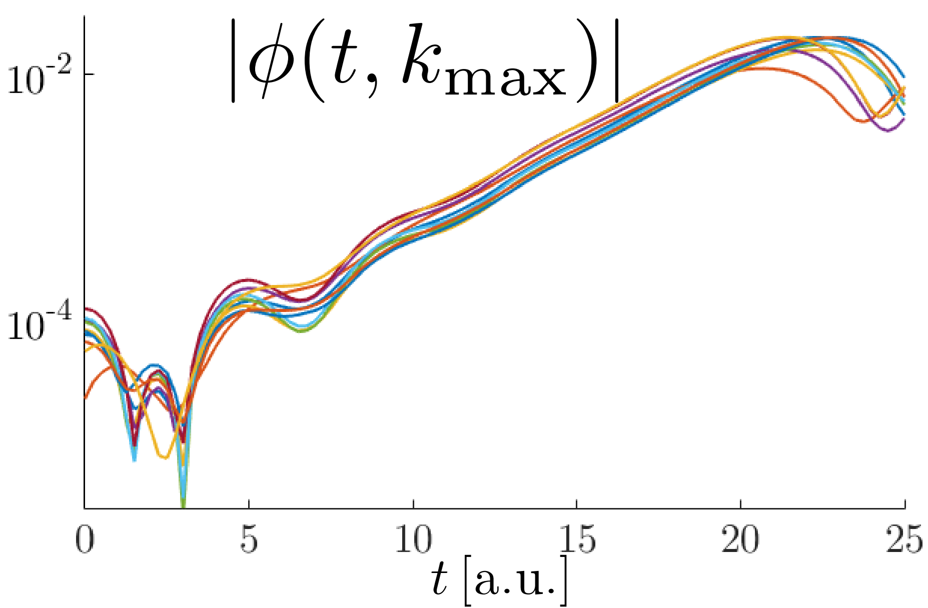
Statistical uncertainty

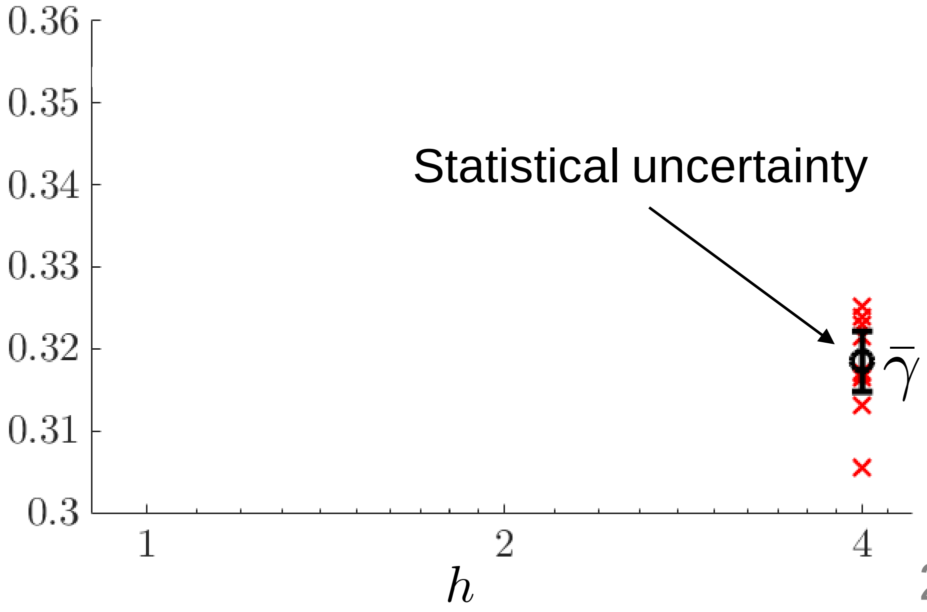
- Choose Δx, Δt, N and perform a simulation
- Repeat with different seeds



Statistical uncertainty

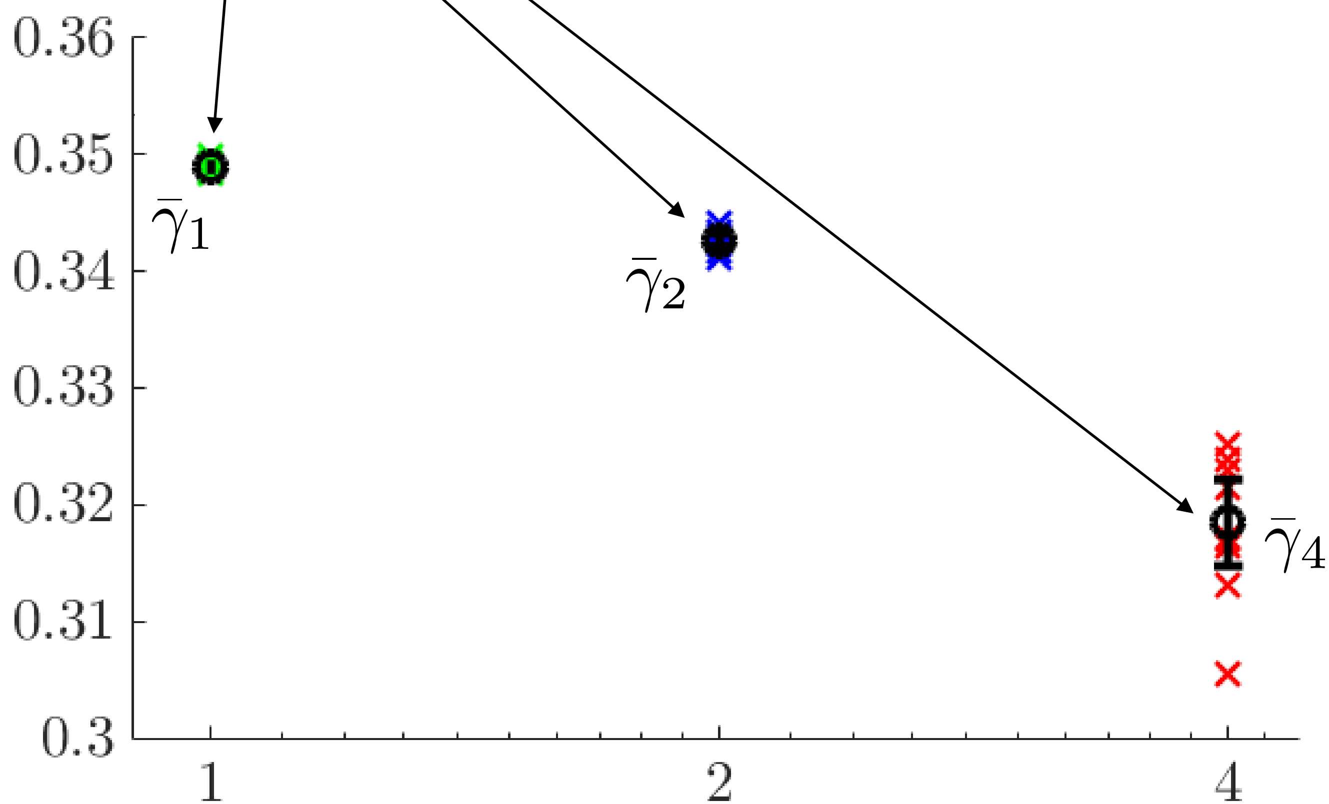
- Choose Δx, Δt, N and perform a simulation
- Repeat with different seeds





Discretization uncertainty

- Choose $\Delta x,\ \Delta t,\ N$ and perform a simulation
- Repeat with different $\Delta x,\ \Delta t,\ N$



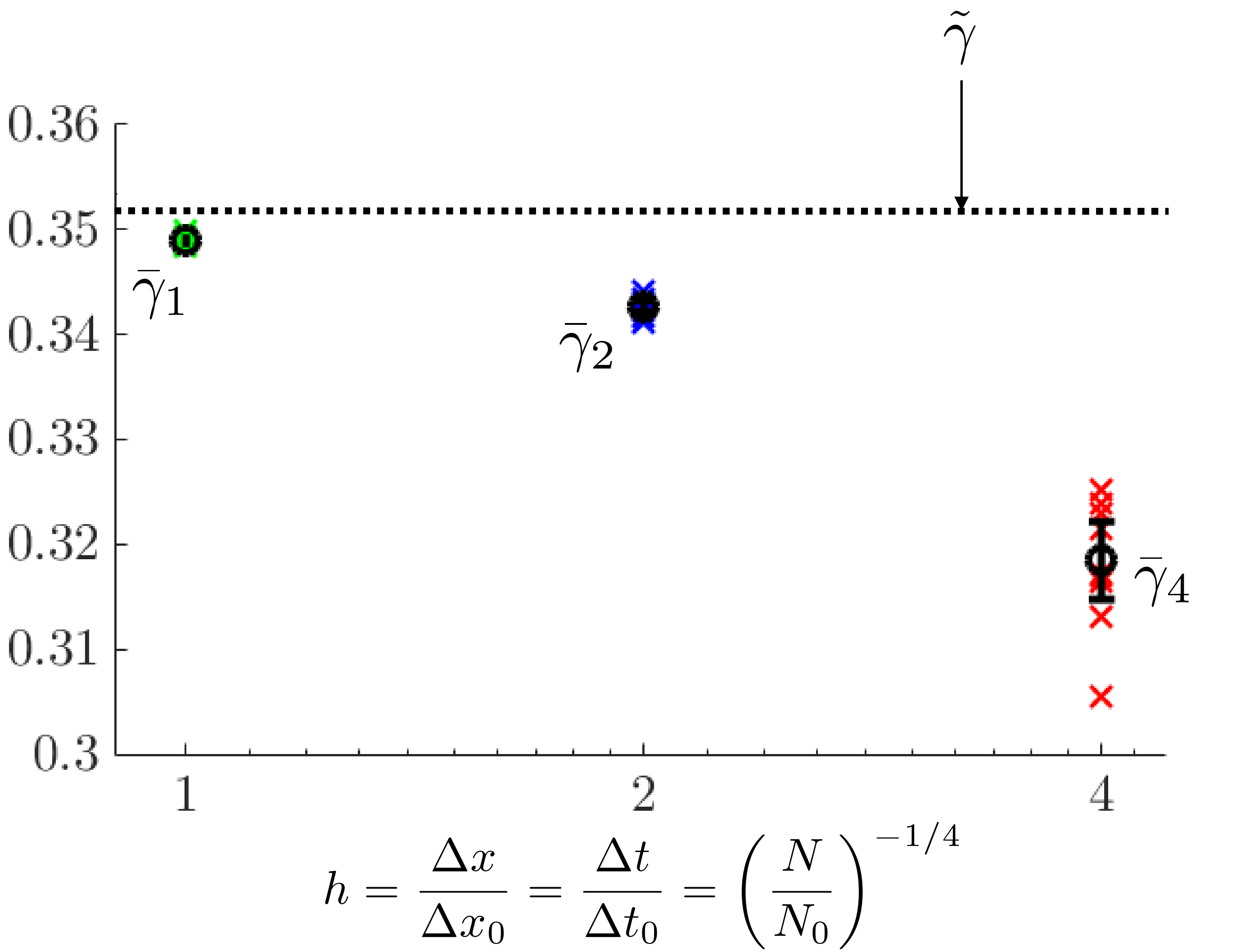
$$h = \frac{\Delta x}{\Delta x_0} = \frac{\Delta t}{\Delta t_0} = \left(\frac{N}{N_0}\right)^{-1/4}$$

Discretization uncertainty

- Choose Δx, Δt, N and perform a simulation
- Repeat with different Δx, Δt, N

Use of high order estimate (Richardson extrapolation)

$$\tilde{\gamma} = \bar{\gamma}_1 + \frac{\bar{\gamma}_1 - \bar{\gamma}_2}{2^p - 1}$$

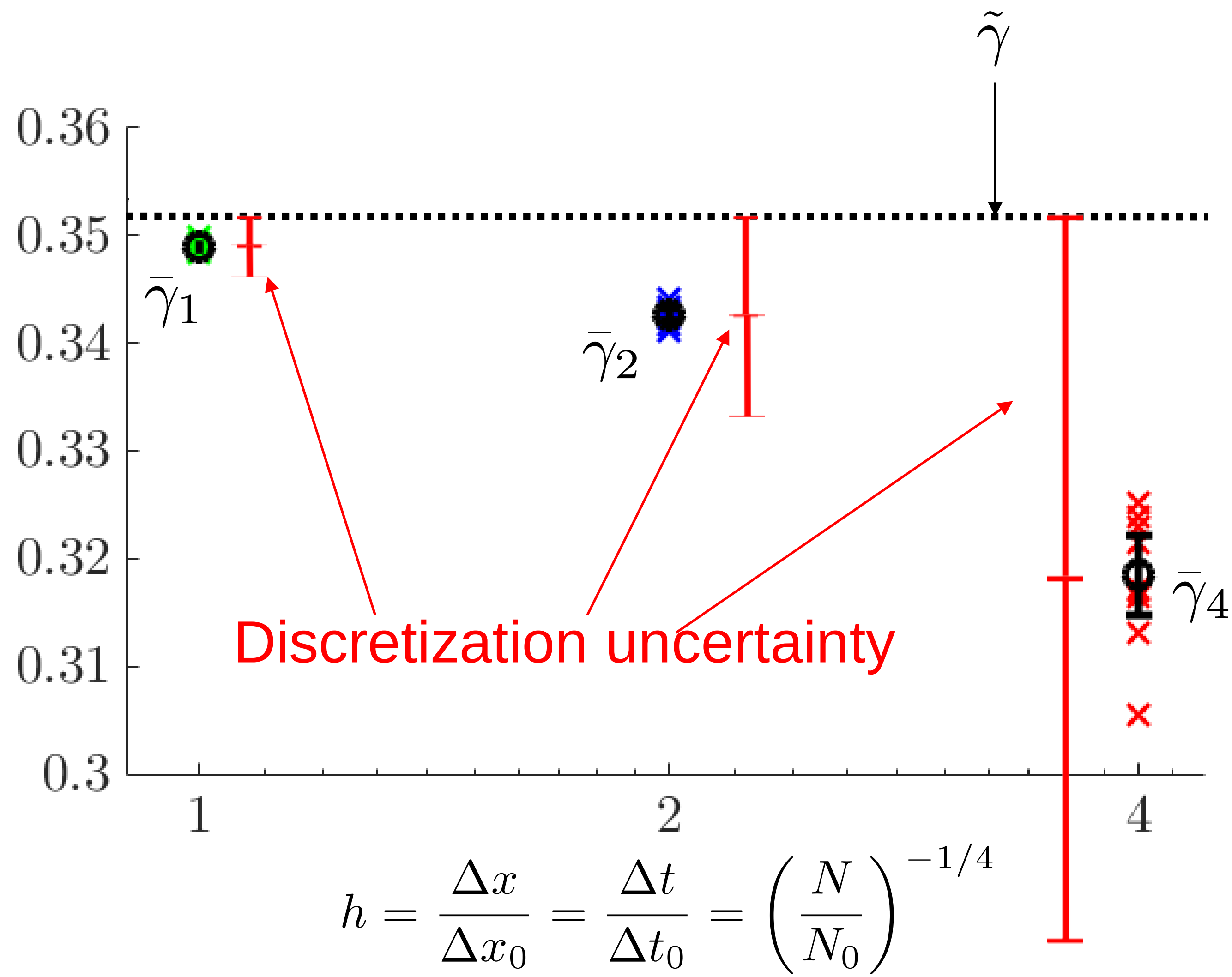


Discretization uncertainty

- Choose Δx, Δt, N and perform a simulation
- Repeat with different Δx, Δt, N

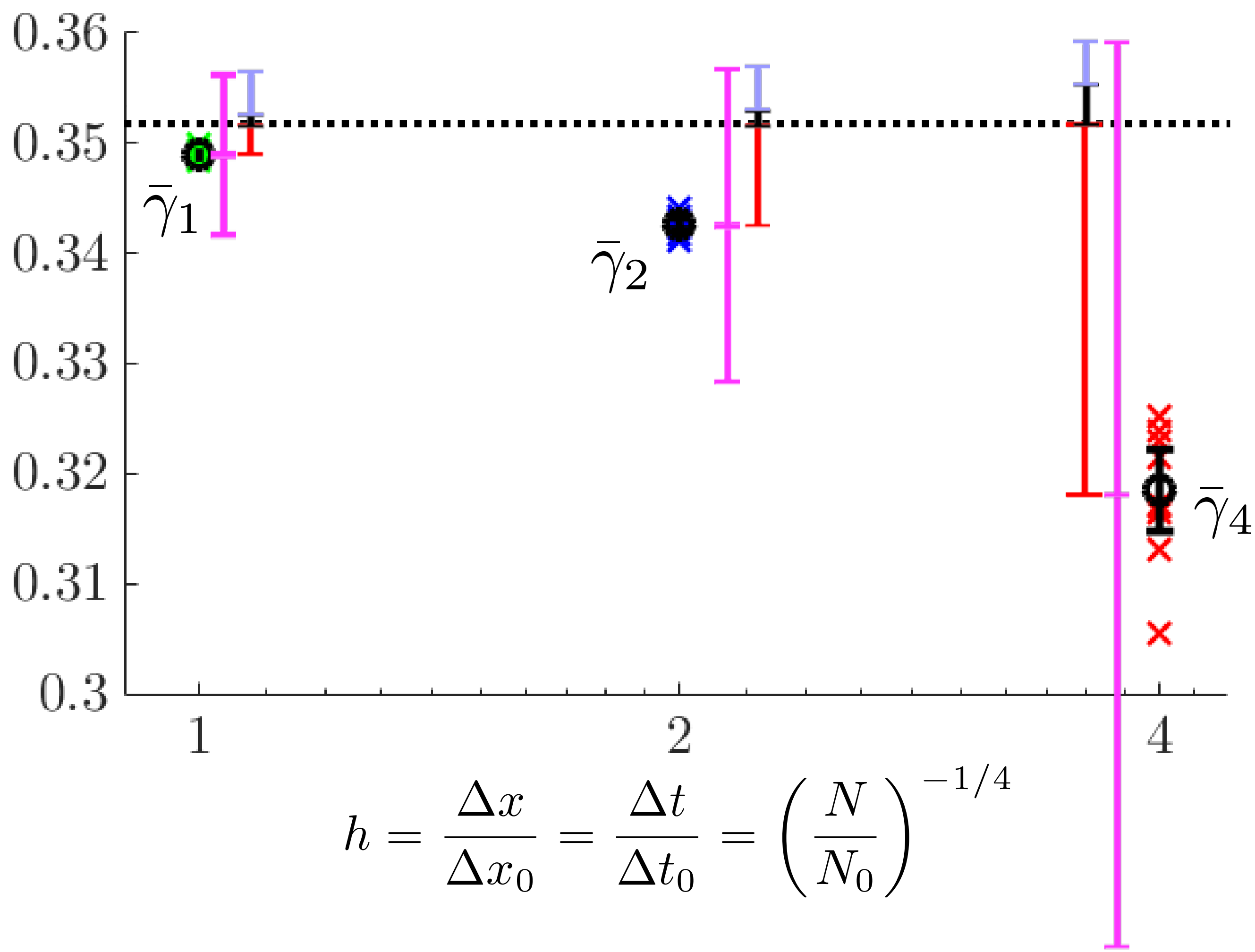
Use of high order estimate (Richardson extrapolation)

$$\tilde{\gamma} = \bar{\gamma}_1 + \frac{\bar{\gamma}_1 - \bar{\gamma}_2}{2^p - 1}$$



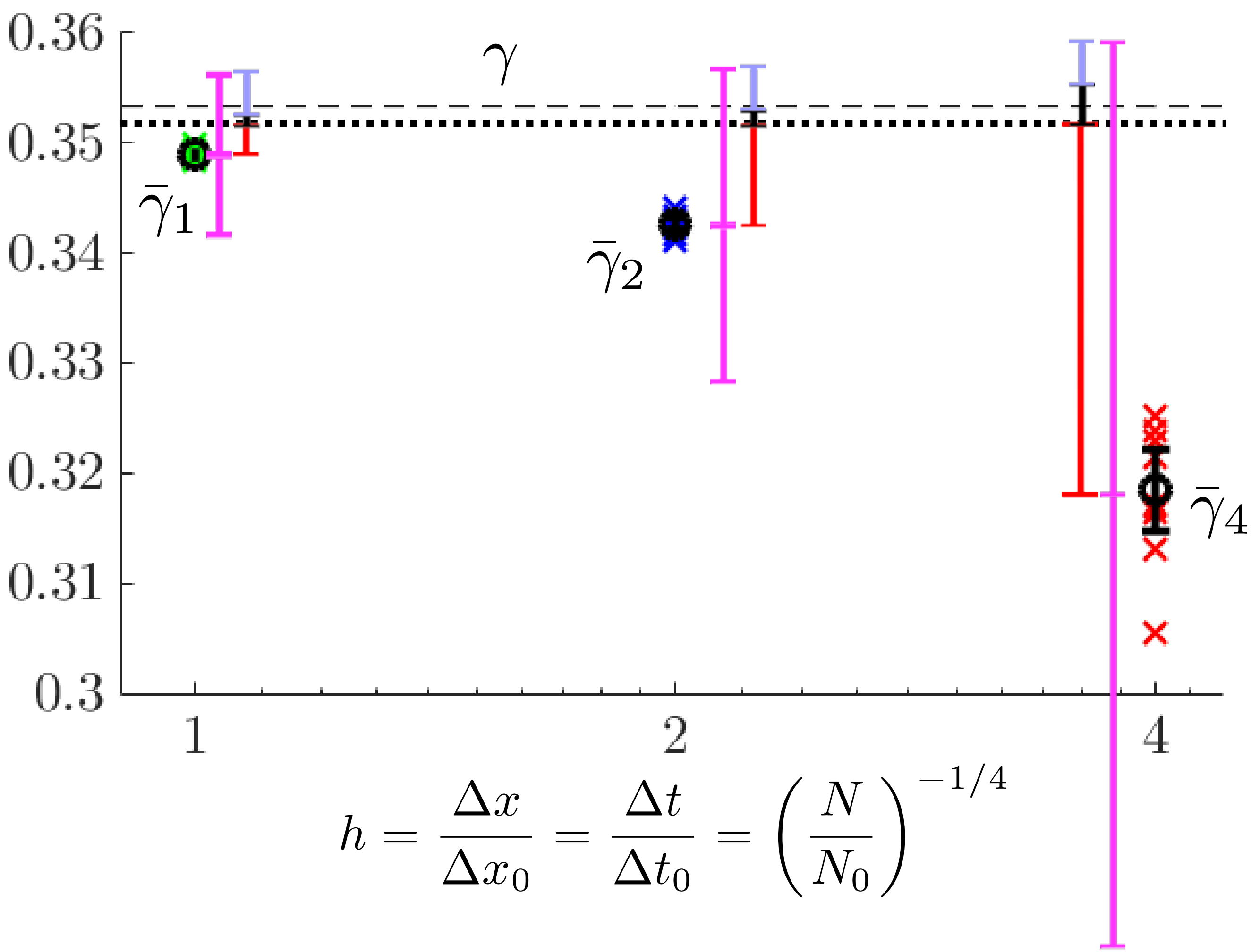
Numerical uncertainty

Numerical uncertainty = post-processing + statistical + discretization



Numerical uncertainty

Numerical uncertainty = post-processing + statistical + discretization



Conclusions

- We provided rigorous methodologies to verify plasma simulations, a crucial issue in plasma physics
- MMS is a methodology now routinely used to rigorously verify plasma simulation codes based on finite differences schemes
- Overcoming the difficulty of comparing distribution functions with markers affected by statistical noise, we now generalized MMS to PIC codes verification
- We provided a methodology to rigorously estimate the uncertainties affecting simulation results due to finite statistics and discretization

More details in [Riva *et al.*, PoP (2014); Riva *et al.*, PoP (2017)]

Open questions

- MMS with shocks and discontinuities
- MMS for simulation codes involving adaptive mesh refinements
- Uncertainty propagation

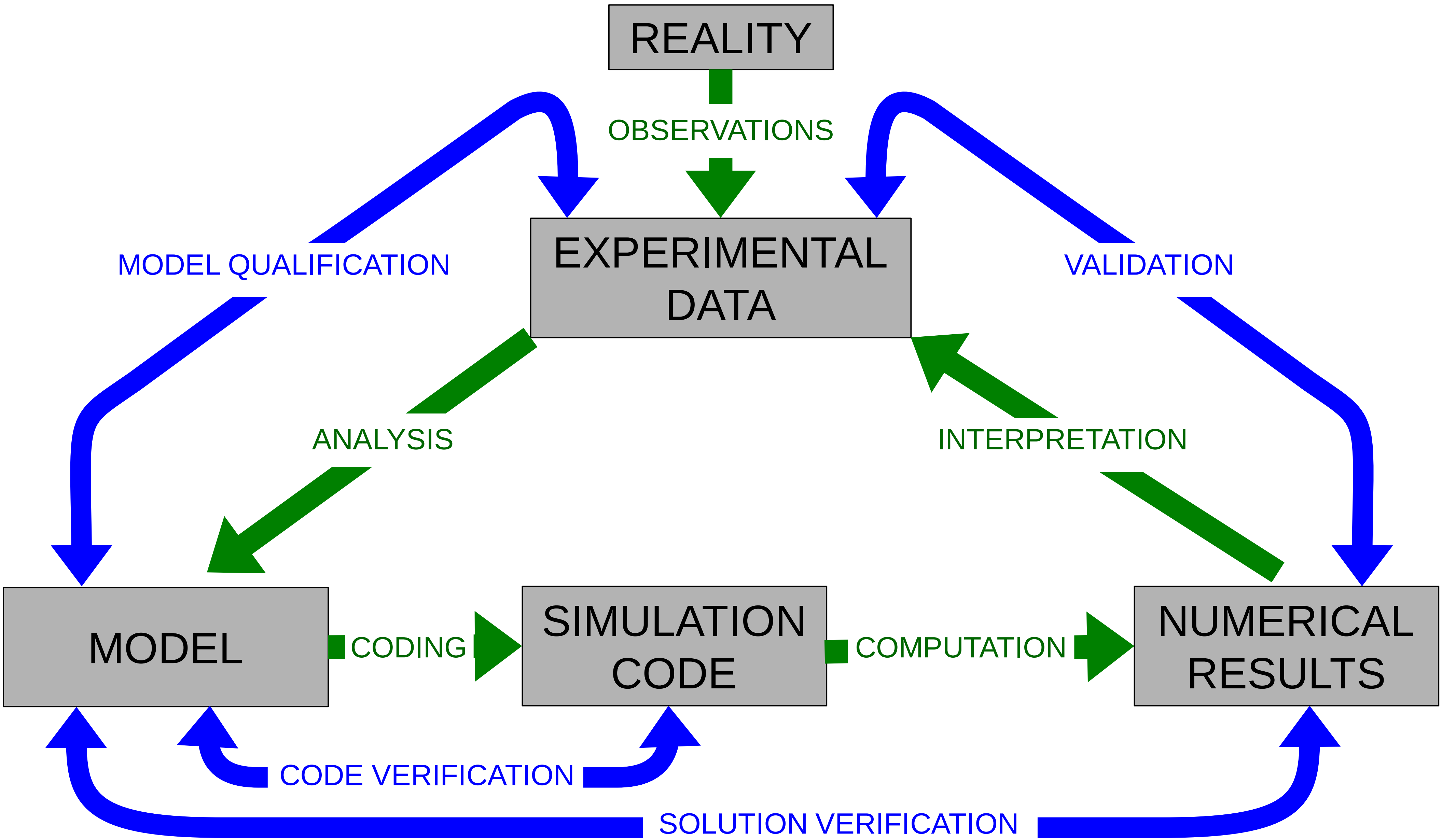
Conclusions

- We provided rigorous methodologies to verify plasma simulations, a crucial issue in plasma physics
- MMS is a methodology now routinely used to rigorously verify plasma simulation codes based on finite differences schemes
- Overcoming the difficulty of comparing distribution functions with markers affected by statistical noise, we now generalized MMS to PIC codes verification
- We provided a methodology to rigorously estimate the uncertainties affecting simulation results due to finite statistics and discretization

More details in [Riva *et al.*, PoP (2014); Riva *et al.*, PoP (2017)]

Backup slides

Verification & Validation

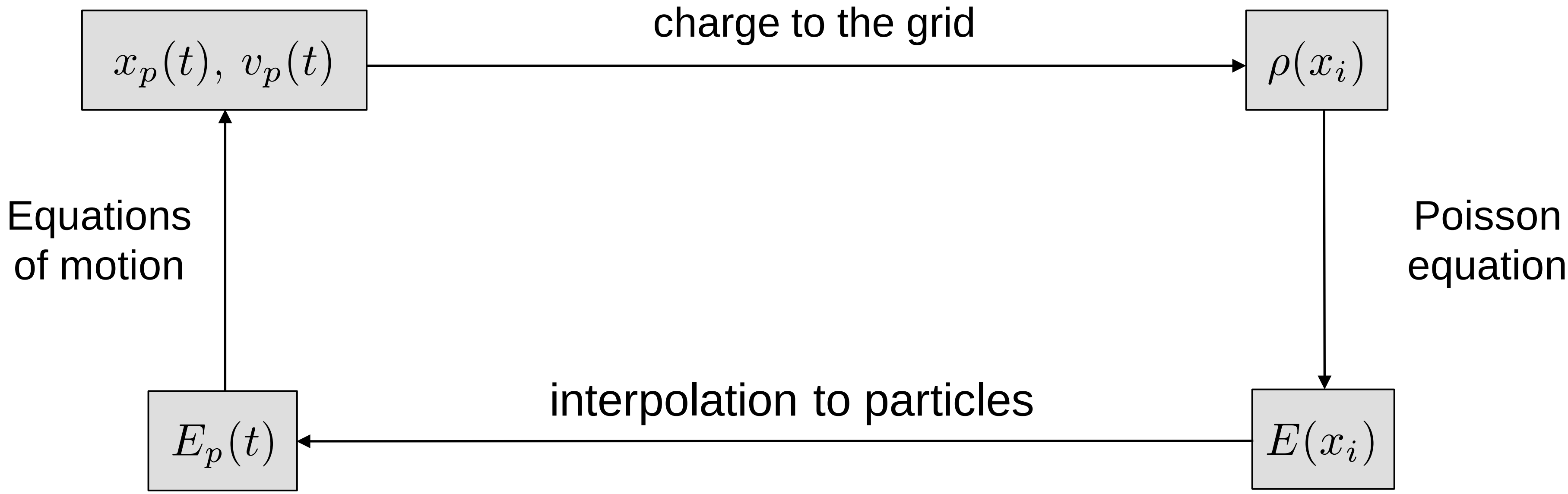


The PIC algorithm

A simple model:

$$\frac{\partial f}{\partial t} + v \cdot \frac{\partial f}{\partial x} + \frac{q}{m} E \cdot \frac{\partial f}{\partial v} = 0$$

$$\frac{\partial E}{\partial x} = \frac{\rho}{\epsilon_0}$$



$$f_N(x, v, t) = \sum_{p=1}^{N} \delta\left[x - x_p(t)\right] \delta\left[v - v_p(t)\right]$$

MMS for a PIC simulation code

The modified model:

$$\frac{\partial f_m}{\partial t} + v \cdot \frac{\partial f_m}{\partial x} + \frac{q}{m} E_m \cdot \frac{\partial f_m}{\partial v} = S_f$$

$$\frac{\partial E_m}{\partial x} = \frac{\rho}{\epsilon_0} + S_E$$

$$\begin{tikzcd}
x_p(t),\, v_p(t),\, w_p(t) \arrow[rr, "\text{charge to the grid}"] & & \rho(x_i) \arrow[d, "\text{Poisson equation}"] \\
E_p(t) \arrow[u, "\text{Equations of motion}"] & & E(x_i) \arrow[ll, "\text{interpolation to particles}"]
\end{tikzcd}$$

$$\frac{dw_p}{dt} = \frac{S_f\left[x_p(t), v_p(t), t\right]}{f_m\left[x_p(0), v_p(0), 0\right]}$$

$$f_N(x, v, t) = \sum_{p=1}^{N} w_p(t) \delta \left[x - x_p(t)\right] \delta \left[v - v_p(t)\right]$$

Implementing MMS in PIC codes

Manufactured solution: $f_M(x,v,t), \phi_M(x,t)$

Source terms:
$$\begin{cases} S_\phi(x,t) = \partial_x^2 \phi + \frac{q}{\epsilon_0} \int_{-\infty}^{+\infty} f_M dv \\ S_f(x,v,t) = \frac{\partial f_M}{\partial t} + v \cdot \frac{\partial f_M}{\partial x} - \frac{q}{m} \frac{\partial \phi}{\partial x} \cdot \frac{\partial f_M}{\partial v} \end{cases}$$

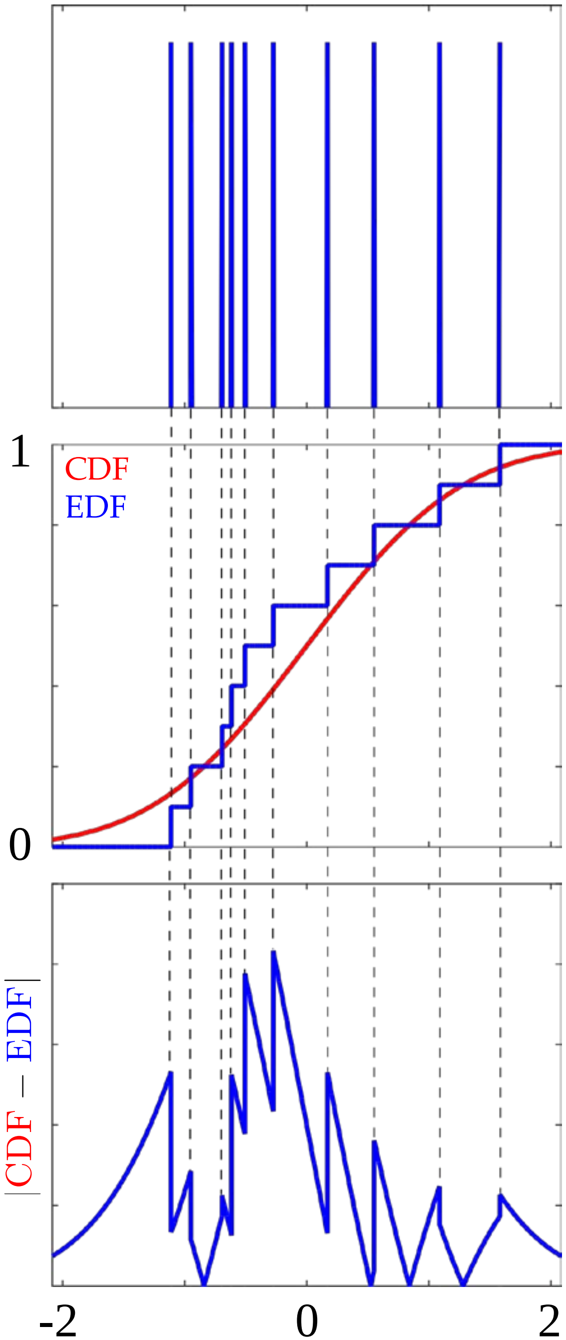
Poisson's equation:
$$\frac{\partial^2 \phi}{\partial x^2}(x,t) = -\frac{\rho\,(x,t)}{\epsilon_0} + S_\phi\,(x,t)$$

Weighted function:
$$f_N(x,v,t) = \sum_{p=1}^{N} w_p(t) \delta\left[x - x_p(t)\right] \delta\left[v - v_p(t)\right]$$

Initial condition: $f_M(x,v,t=0) = f_0(x,v) \cdot w(x,v)$

Equations of motion:
$$\begin{cases} \frac{dw_p}{dt} = \frac{S_f[x_p(t), v_p(t), t]}{f_0[x_p(0), v_p(0)]} \\ \frac{dx_p}{dt} = v_p(t) \\ \frac{dv_p}{dt} = \frac{q}{m} E_p(t) \end{cases}$$

The Kolmogorov–Smirnov statistic



CDF: $F(x) = P(X \le x) = \int_{-\infty}^{x} f(x')dx'$

Indicator function: $I_A(a) = \begin{cases} 1 & \text{if} \quad a \in A \\ 0 & \text{if} \quad a \notin A \end{cases}$

EDF: $F_N(x) = \frac{1}{N} \sum_{i=1}^{N} I_{]-\infty,x]}(X_i)$

KS statistic: $D_N = \sup_{x \in \mathbb{R}} |F(x) - F_N(x)|$

Under null hypothesis:

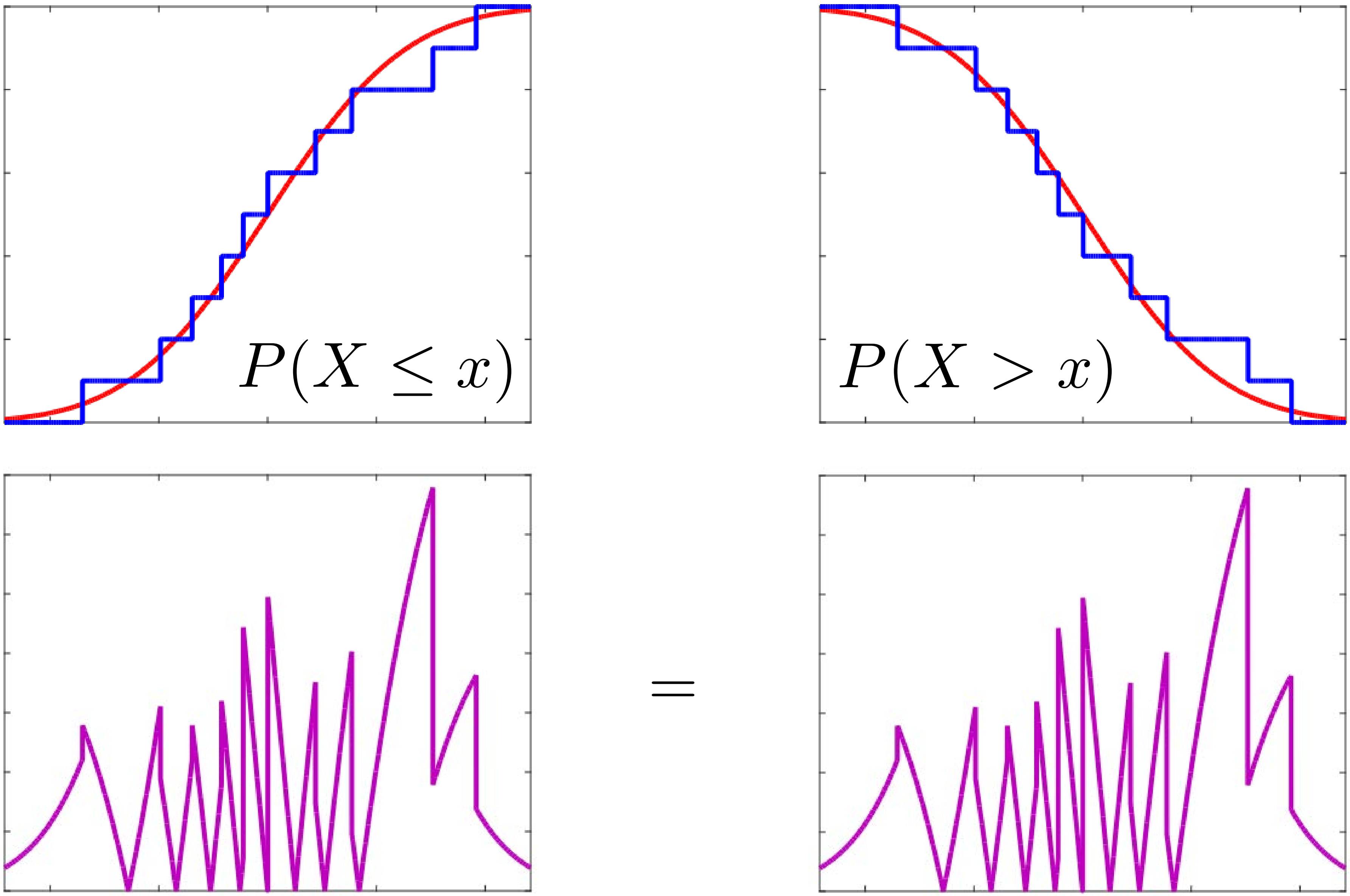
$D_N \xrightarrow{\text{almost surely}} 0$ and $\sqrt{N} D_N \xrightarrow{N \to \infty} \sup_{x \in \mathbb{R}} |B(F(x))|$

where $B(t)$ is the Brownian bridge

$$D_N = \sup_{x \in \mathbb{R}} |P(x \le X) - F_N(x)| = \sup_{x \in \mathbb{R}} |P(X > x) - [1 - F_N(x)]\,|$$

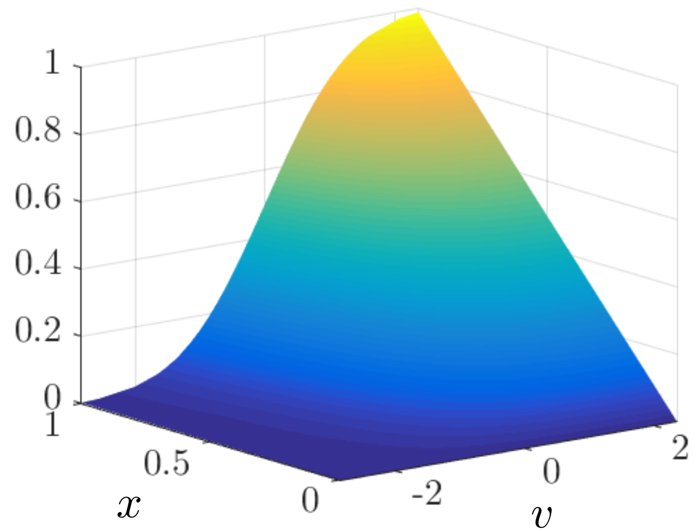
A useful property

1D independent of integration direction:

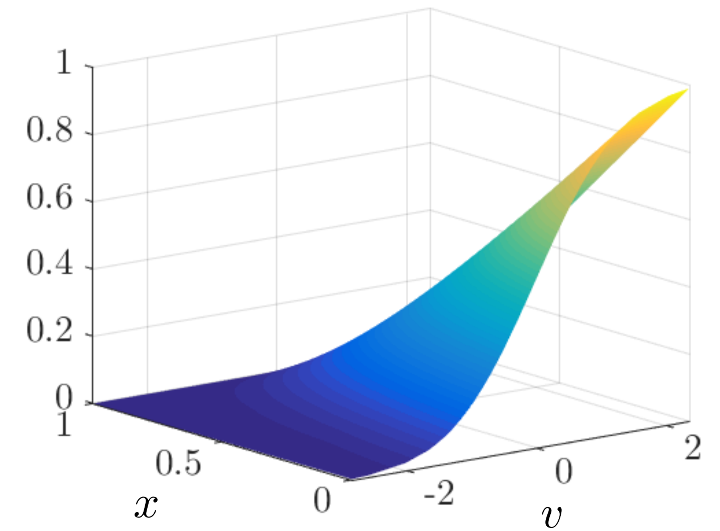


2D cumulative distribution functions

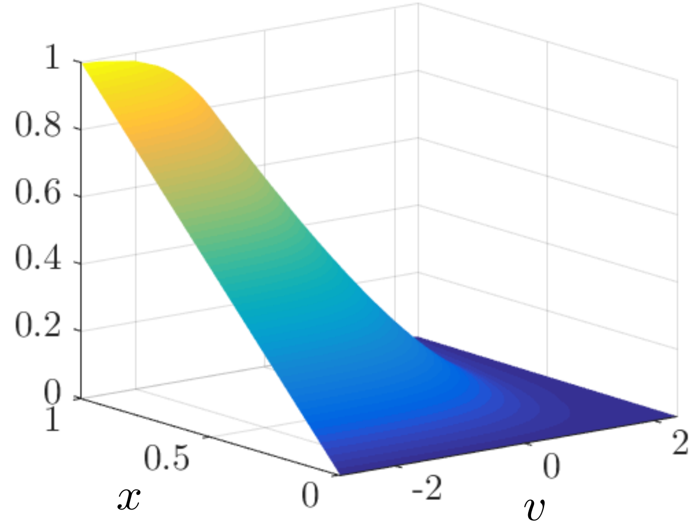
$$P(X \le x, V \le v) = \int_{-\infty}^{x} dx' \int_{-\infty}^{v} dv' f(x', v')$$



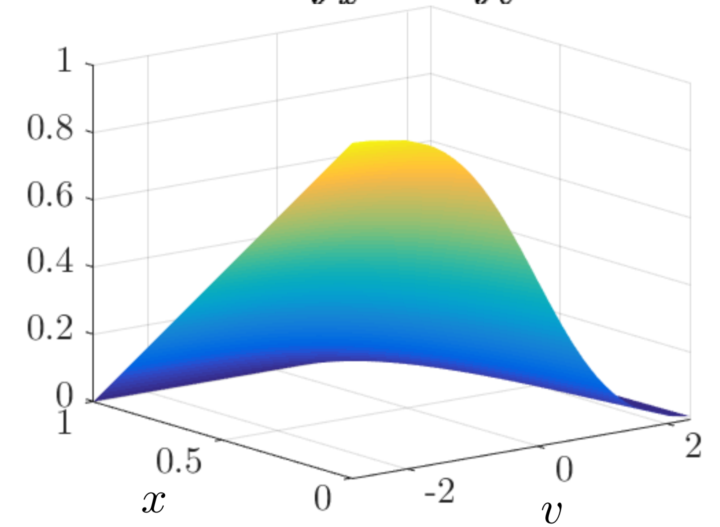
$$P(X > x, V \le v) = \int_{x}^{\infty} dx' \int_{-\infty}^{v} dv' f(x', v')$$



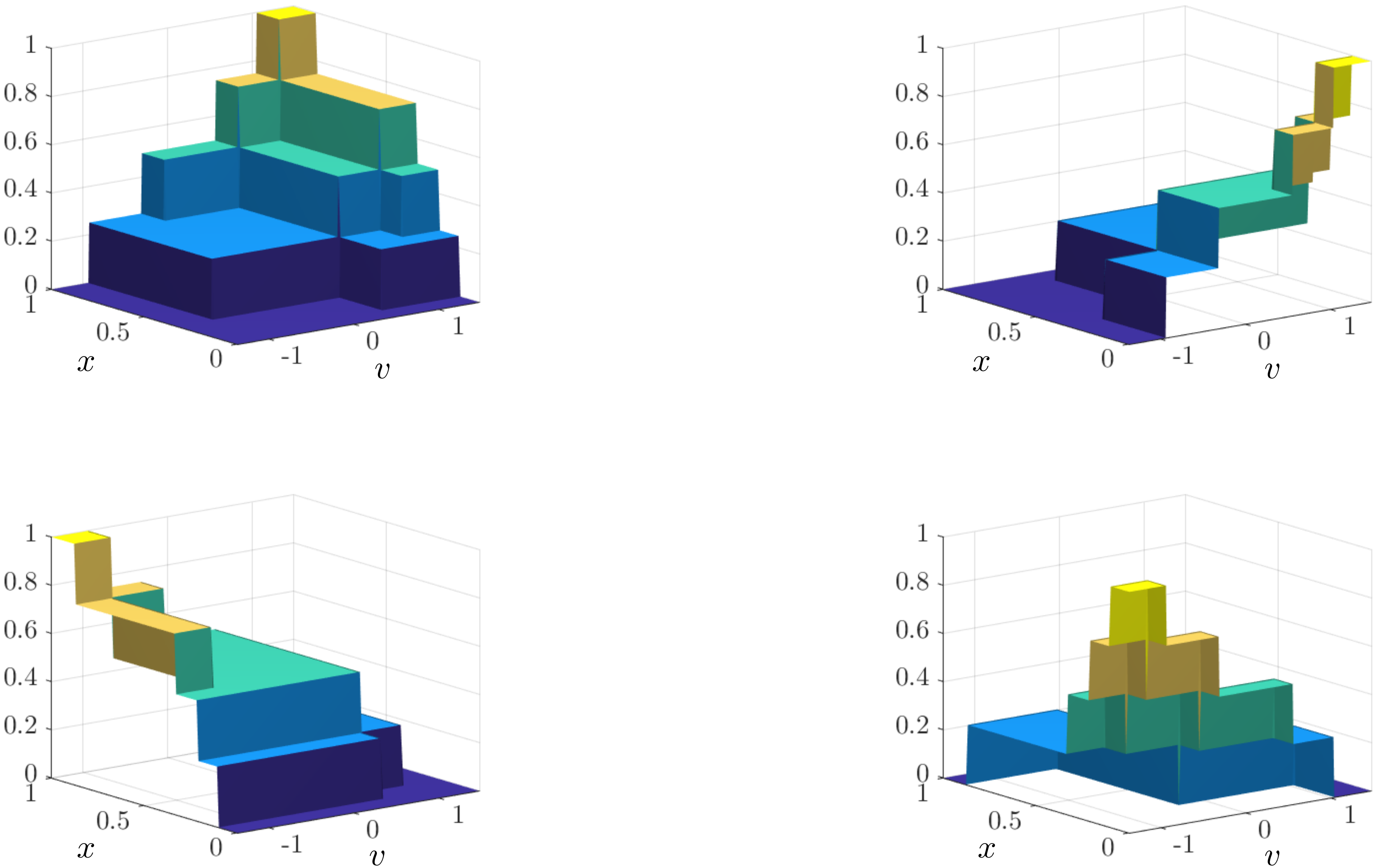
$$P(X \le x, V > v) = \int_{-\infty}^{x} dx' \int_{v}^{\infty} dv' f(x', v')$$



$$P(X > x, V > v) = \int_{x}^{\infty} dx' \int_{v}^{\infty} dv' f(x', v')$$

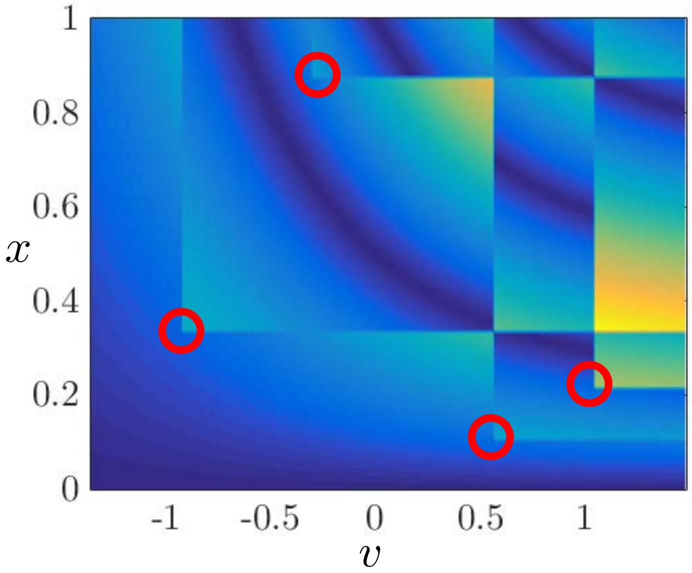


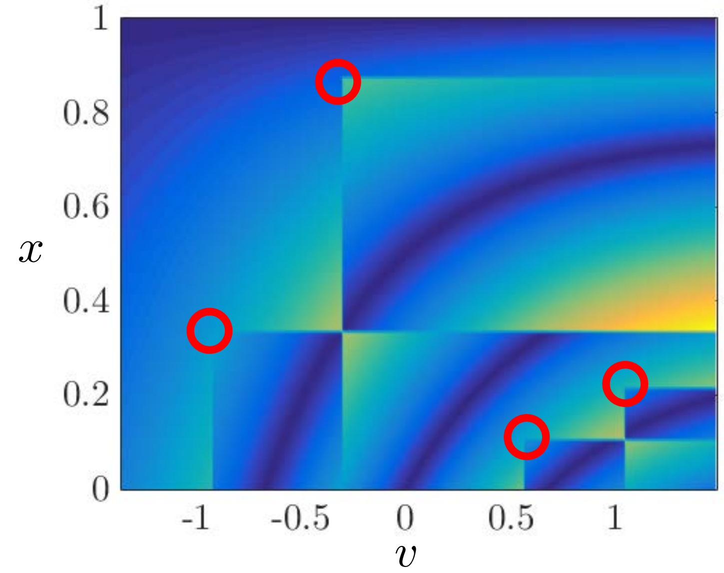
2D empirical distribution functions

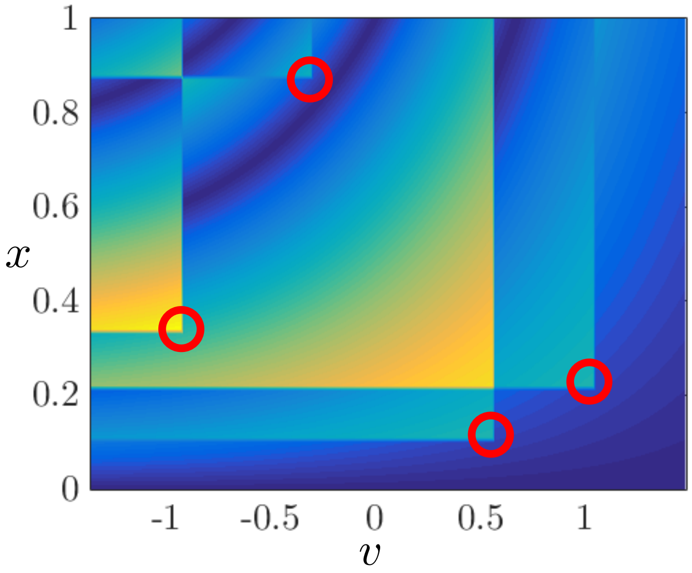


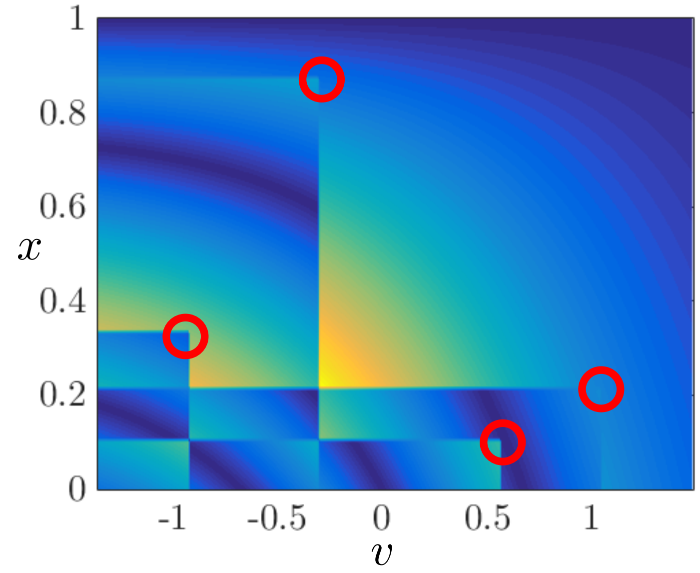
Multidimensional case

$$|\mathrm{CDF} - \mathrm{EDF}|$$









Two-dimensional Peacock test

Generalization of the KS statistic [J.A. Peacock 1983]:

$$F^1(x,y) = \int_{-\infty}^{x}\int_{-\infty}^{y} f(x',y')dx'dy' \qquad F_N^1(x,y) = \frac{1}{N}\sum_{i=1}^{N} I_{]-\infty,x]}(X_i)\, I_{]-\infty,y]}(Y_i)$$

$$F^2(x,y) = \int_{x}^{+\infty}\int_{-\infty}^{y} f(x',y')dx'dy' \qquad F_N^2(x,y) = \frac{1}{N}\sum_{i=1}^{N} I_{]x,+\infty[}(X_i)\, I_{]-\infty,y]}(Y_i)$$

$$F^3(x,y) = \int_{x}^{+\infty}\int_{y}^{+\infty} f(x',y')dx'dy' \qquad F_N^3(x,y) = \frac{1}{N}\sum_{i=1}^{N} I_{]x,+\infty[}(X_i)\, I_{]y,+\infty[}(Y_i)$$

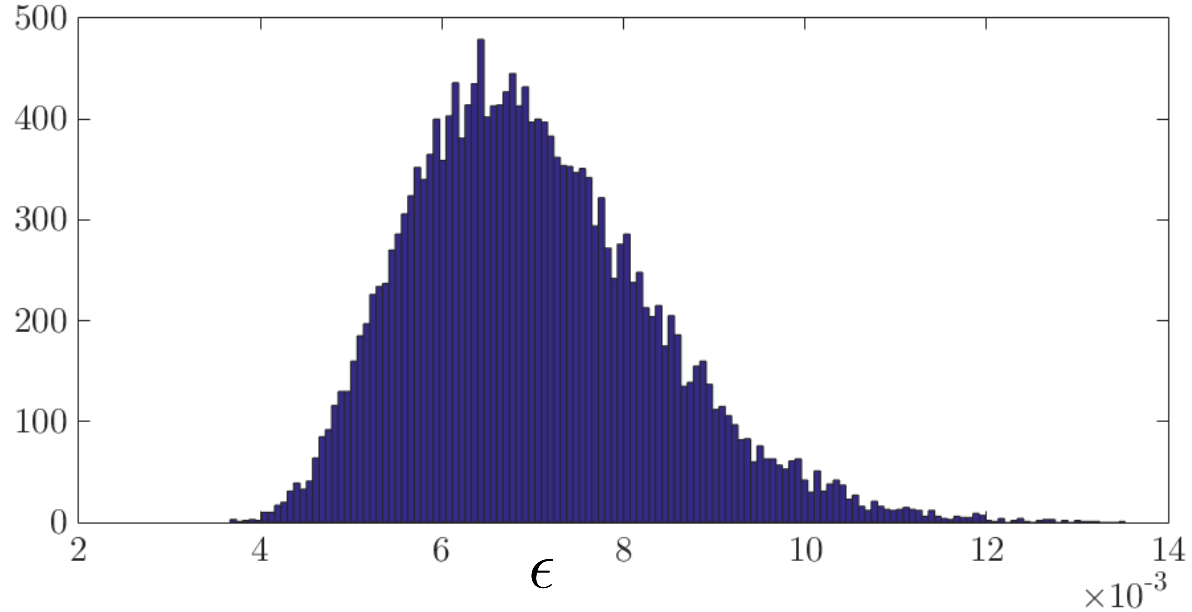
$$F^4(x,y) = \int_{-\infty}^{x}\int_{y}^{+\infty} f(x',y')dx'dy' \qquad F_N^4(x,y) = \frac{1}{N}\sum_{i=1}^{N} I_{]-\infty,x]}(X_i)\, I_{]y,+\infty[}(Y_i)$$

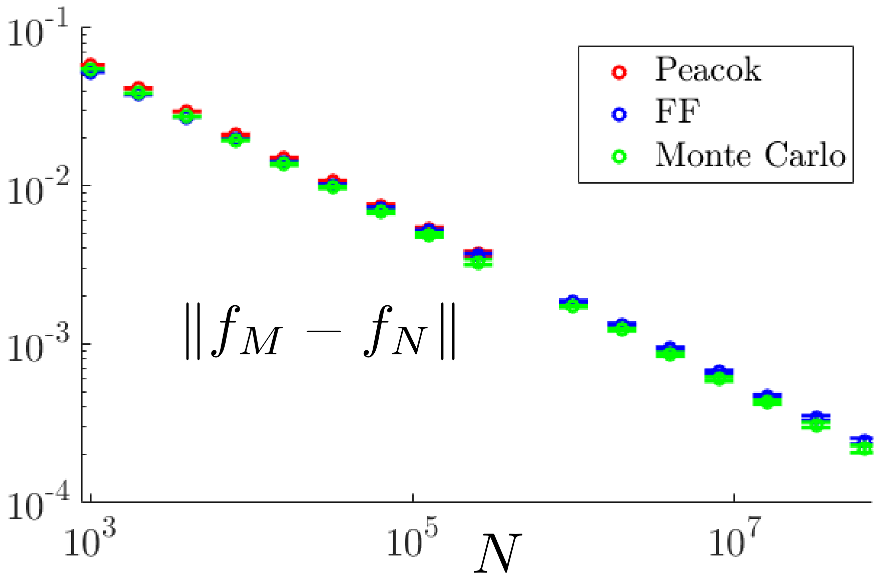
$$D_N^k = \sup_{(x,y)\in\mathbb{R}} \left|F^k(x,y) - F_N^k(x,y)\right| \qquad k = 1,2,3,4$$

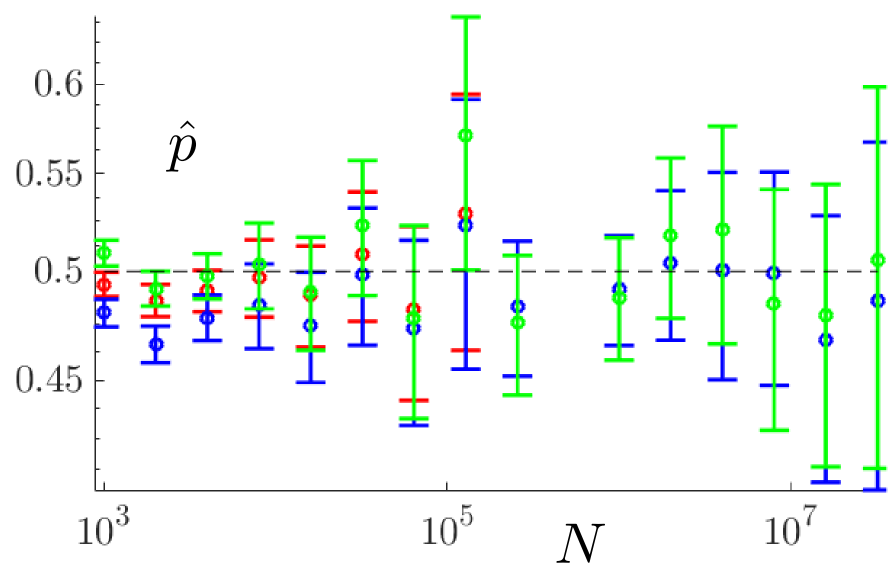
$$\boxed{D_N = \max\left(D_N^1, D_N^2, D_N^3, D_N^4\right)}$$

$$I_A(a) = \begin{cases} 1 & \text{if} \quad a \in A \\ 0 & \text{if} \quad a \notin A \end{cases}$$

Different approaches







Numerical scheme

- Interpolation scheme: first-order weighting (CIC PIC)
- Interpolation function: $$I(x) = \begin{cases} 0 & \text{if } |x| > \Delta x \\ \frac{x}{\Delta x} + 1 & \text{if } -\Delta x \leq x < 0 \\ -\frac{x}{\Delta x} + 1 & \text{if } 0 \leq x \leq \Delta x \end{cases}$$
- Interpolation: particles → grid

$$\rho_i^n = \frac{q}{\Delta x} \sum_{p=1}^{N} w_p^n I\left(x_i - x_p^n\right)$$

- Poisson solver: second order centered finite differences

$$\frac{\phi_{i-1}^n - 2\phi_i^n + \phi_{i+1}^n}{\Delta x^2} = -\rho_i^n \qquad E_i^n = \frac{\phi_{i-1}^n - \phi_{i+1}^n}{2\Delta x}$$

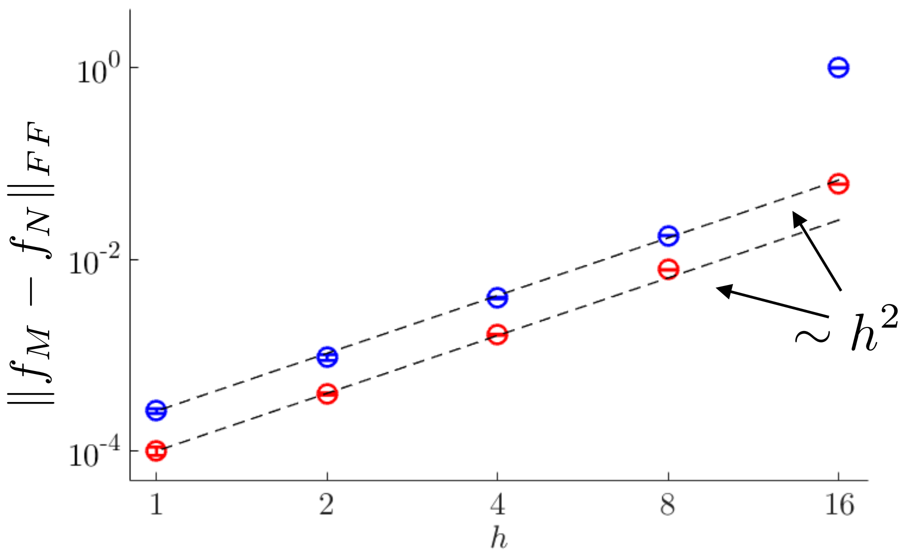
- Interpolation: grid → particles

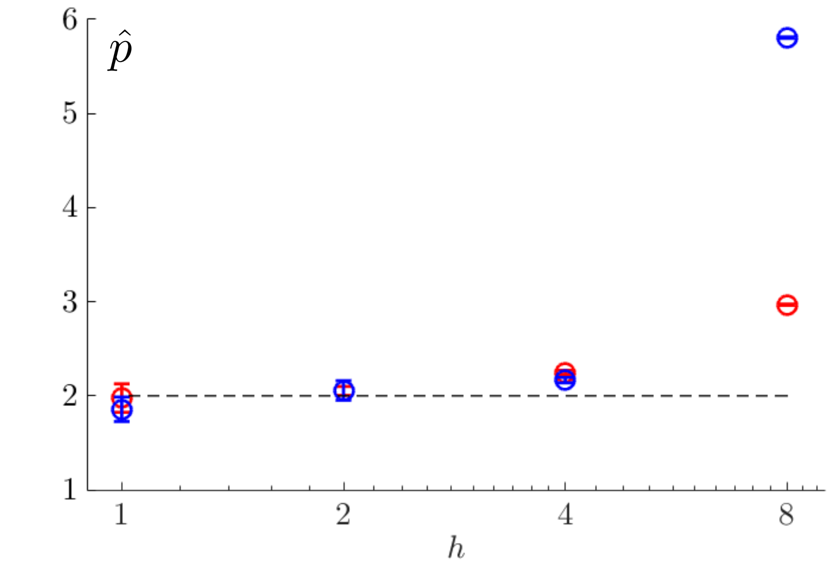
$$E_p^n = \sum_{i=0}^{M} I\left(x_i - x_p^n\right) E_i^n$$

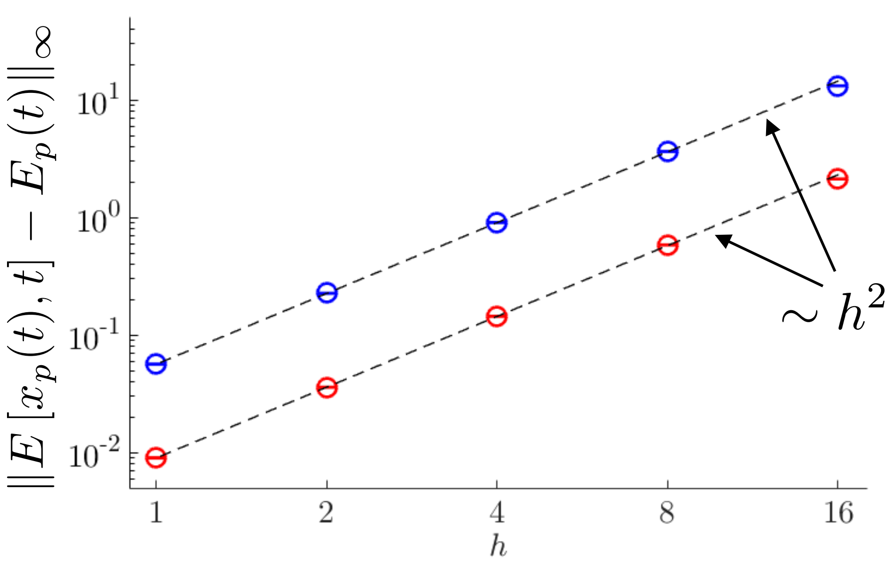
- Time integration: Leapfrog integration scheme

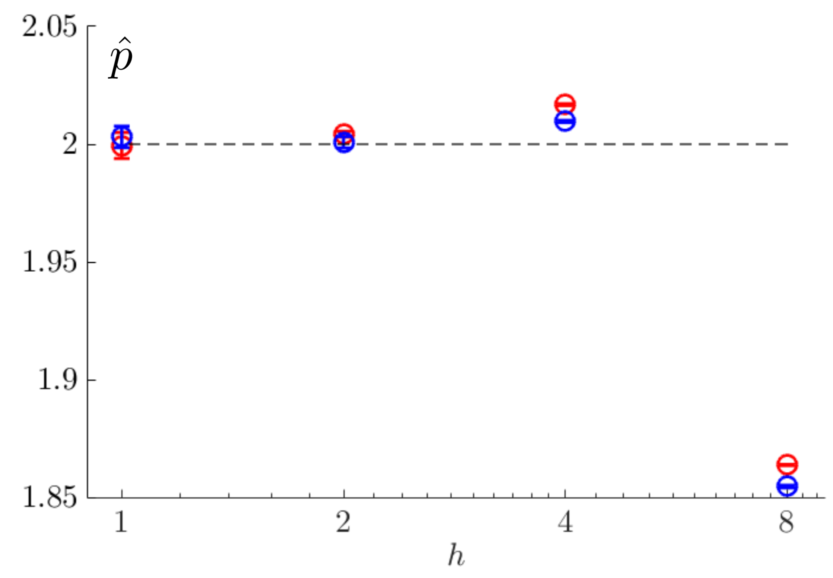
$$\begin{cases} w_p^{n+1} &= w_p^n + \frac{S_f\left[x_p^{n+1/2}, v_p^{n+1/2}, \left(n+\frac{1}{2}\right)\Delta t\right]}{f_0\left(x_p^0, v_p^0\right)} \Delta t \\ x_p^{n+1} &= x_p^n + v_p^{n+1/2} \Delta t \\ v_p^{n+1/2} &= v_p^{n-1/2} + \frac{q}{m} E_p^n \end{cases}$$

Results: PIC code verification



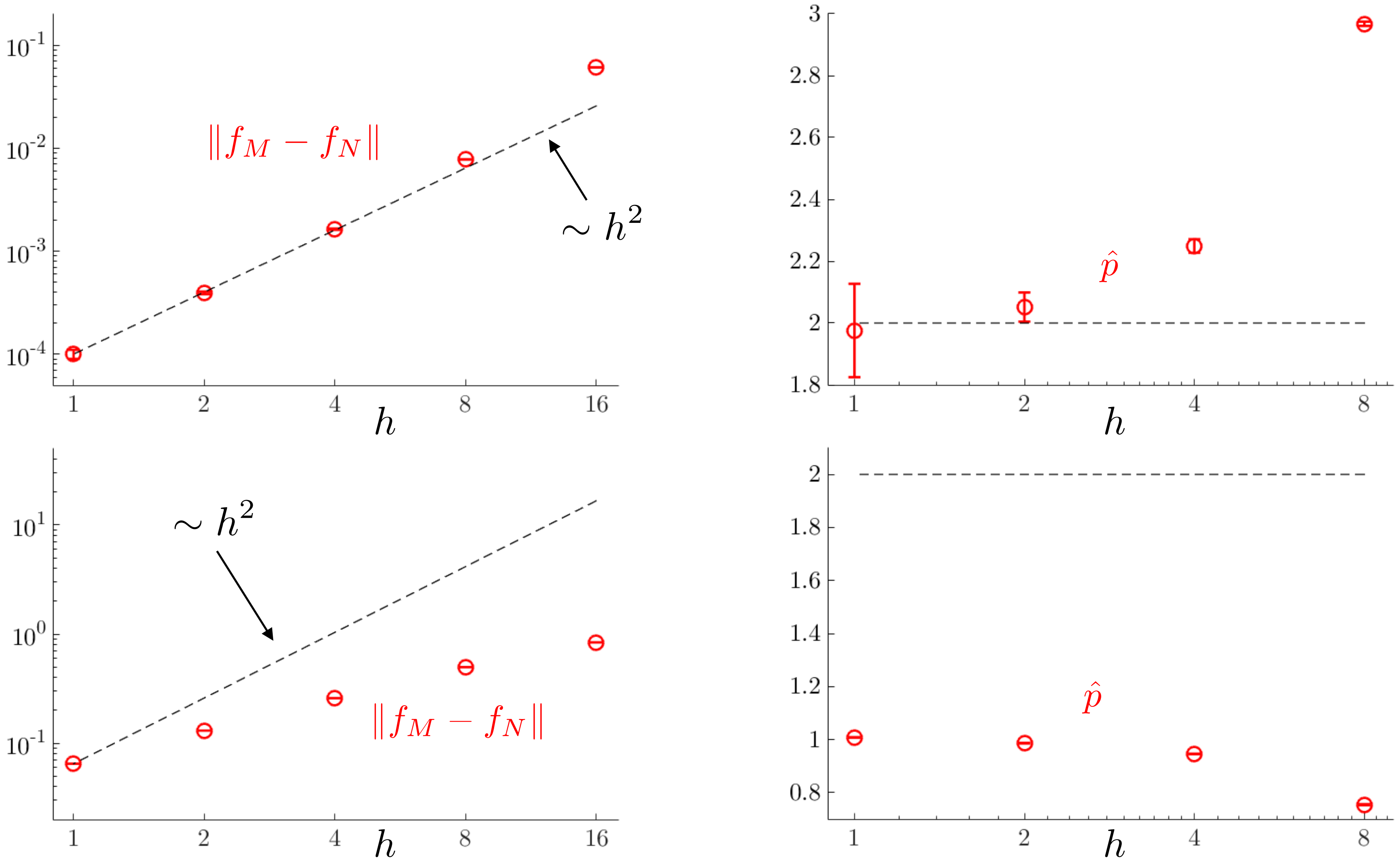






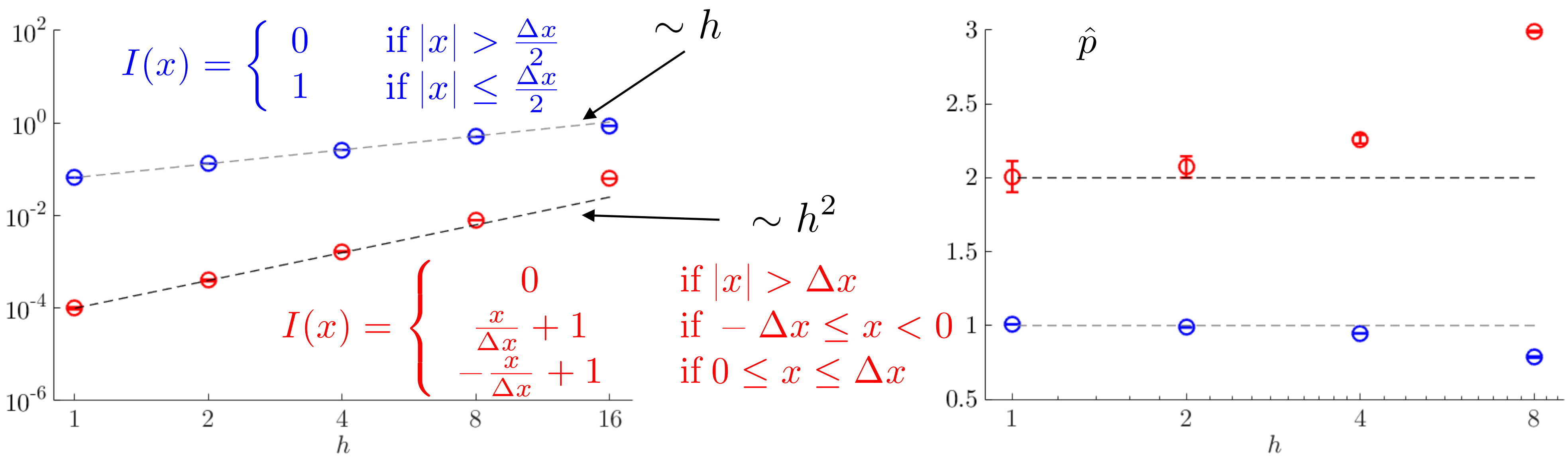
Results: PIC code verification

For $h = \dfrac{\Delta x}{\Delta x_0} = \dfrac{\Delta t}{\Delta t_0} = \left(\dfrac{N}{N_0}\right)^{-1/4}$ we expect $\epsilon = \mathcal{O}\left(h^2\right)$



Monte-Carlo Method

Use Monte-Carlo method to estimate $D_N^k = \sup_{(x,y)\in\mathbb{R}} |F^k(x,y) - F_N^k(x,y)|$

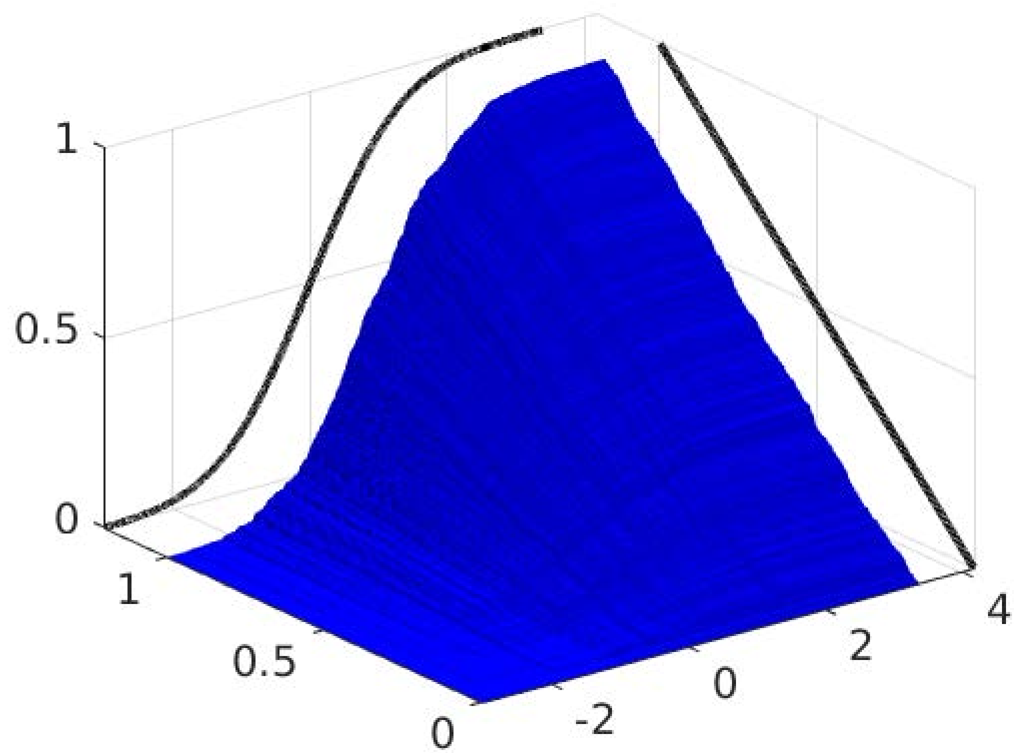


An alternative approach

Decoupling the different dimensions:

$$\epsilon_x(f_M,t) = \sup_{x\in\mathbb{R}} \left| \int_{-\infty}^{x} \left(\int_{-\infty}^{+\infty} f_M(x',v,t)dv \right) dx' - \sum_{p=1}^{N} \hat{w}_p(t) I_{]-\infty,x]}\left[x_p(t)\right] \right|$$

$$\epsilon_v(f_M,t) = \sup_{v\in\mathbb{R}} \left| \int_{-\infty}^{v} \left(\int_{-\infty}^{+\infty} f_M(x,v',t)dx \right) dv' - \sum_{p=1}^{N} \hat{w}_p(t) I_{]-\infty,v]}\left[v_p(t)\right] \right|$$

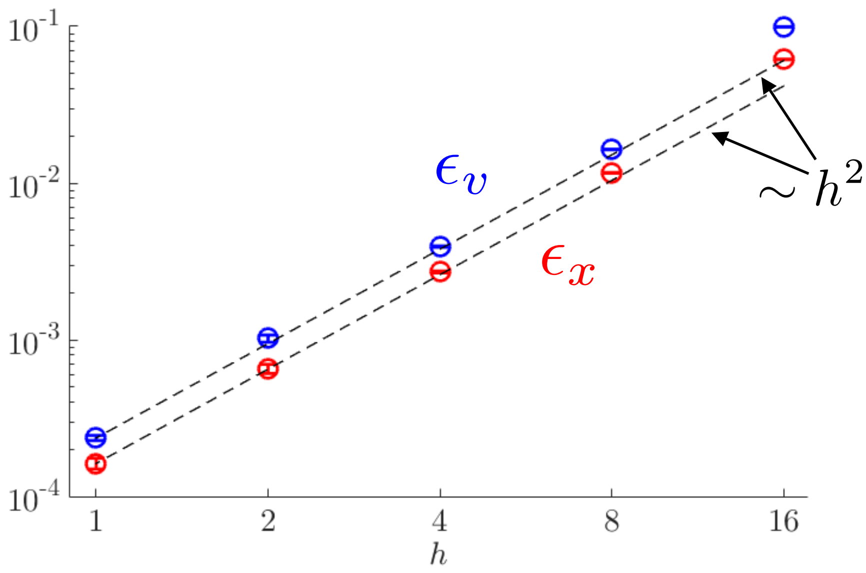


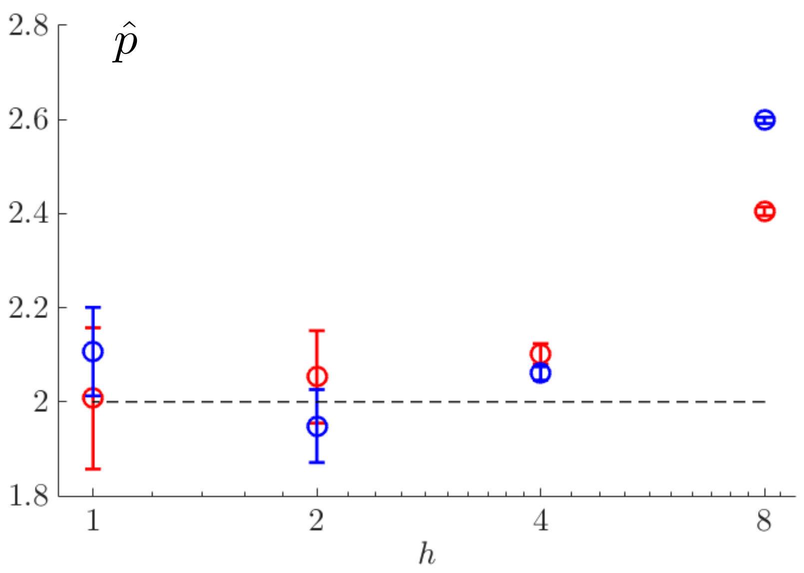
An alternative approach

Decoupling the different dimensions:

$$\epsilon_x(f_M,t) = \sup_{x\in\mathbb{R}} \left| \int_{-\infty}^{x} \left(\int_{-\infty}^{+\infty} f_M(x',v,t)dv \right) dx' - \sum_{p=1}^{N} \hat{w}_p(t) I_{]-\infty,x]}\left[x_p(t)\right] \right|$$

$$\epsilon_v(f_M,t) = \sup_{v\in\mathbb{R}} \left| \int_{-\infty}^{v} \left(\int_{-\infty}^{+\infty} f_M(x,v',t)dx \right) dv' - \sum_{p=1}^{N} \hat{w}_p(t) I_{]-\infty,v]}\left[v_p(t)\right] \right|$$





Statistical errors, the principles

How to estimate the statistical uncertainty affecting a quantity X?

Assumptions:

- X randomly distributed from $f(X)$
- Unknown, finite mean μ_X
- Unknown, finite variance σ_X^2

Perform n_s observations $\Rightarrow X_1, ..., X_{n_s}$

Law of large numbers	$\bar{X}_{n_s} = \frac{1}{n_s} \sum_{i=1}^{n_s} X_i \xrightarrow{n_s \to \infty} \mu_X$
Central limit theorem	$\sqrt{n_s}\left(\bar{X}_{n_s} - \mu_X\right) \xrightarrow{d} N\left(0, \sigma_X^2\right)$

Estimate of statistical uncertainties

Perform n_s simulations with $N' < N$ particles $\Rightarrow X'_1, ..., X'_{n_s}$

Assume $\sigma_X^2 \propto \dfrac{1}{N}$

Estimate X with N particles

$$\Delta X = 1.96 \sqrt{\frac{N'}{N(n_s - 1)} \sum_{i=1}^{n_s} \left[X'_i - \frac{1}{n_s} \sum_{j=1}^{n_s} X'_j \right]^2}$$

For one simulation with N particles: $\Delta X = 1.96 \sigma'_X \sqrt{\dfrac{N'}{N}}$

Statistical error affecting a functional

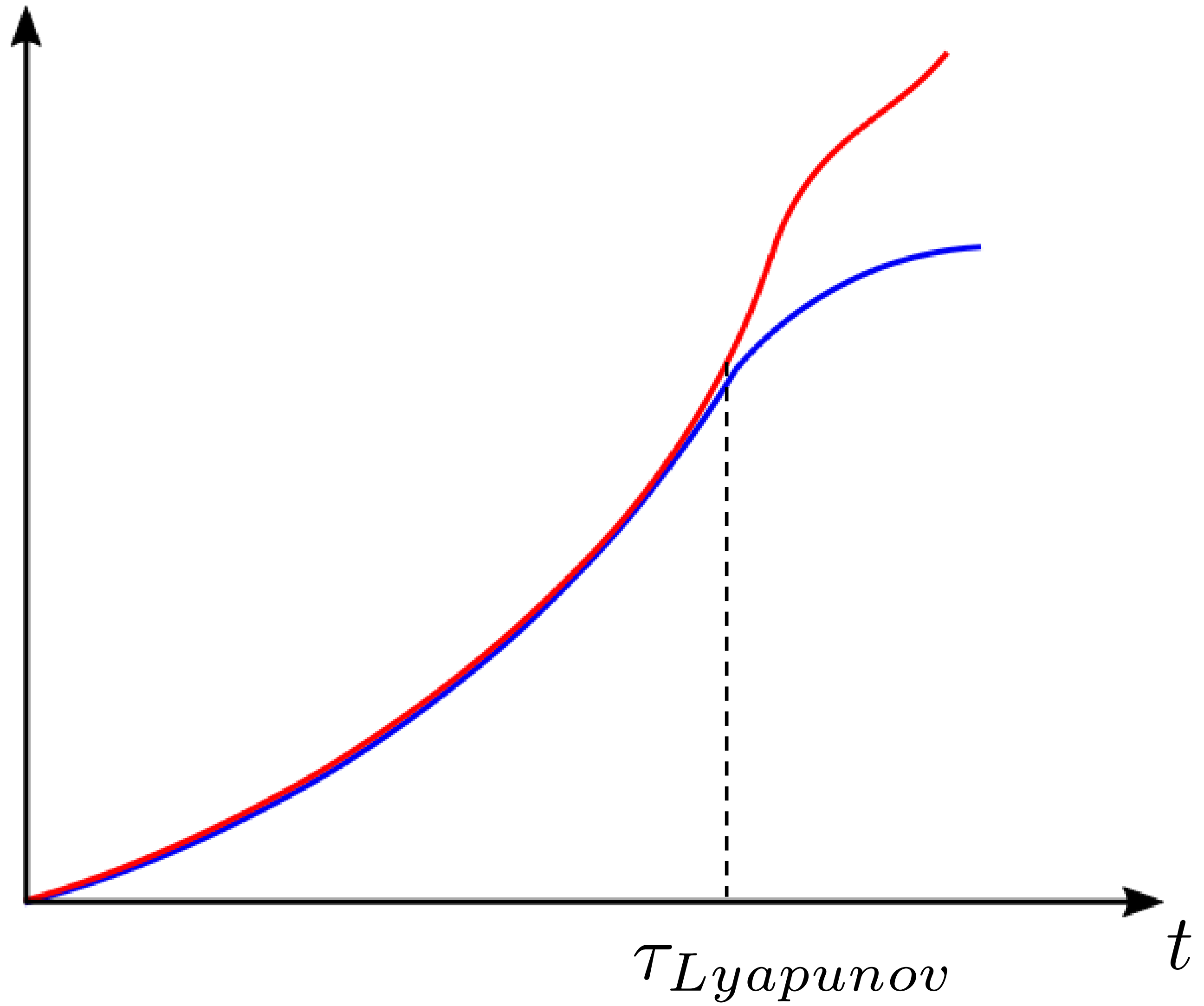
Functional $F(X,Y)$ with uncorrelated X and Y

$$\Delta F = \sqrt{\left(\frac{\partial F}{\partial X}\right)^2 \Delta X^2 + \left(\frac{\partial F}{\partial Y}\right)^2 \Delta Y^2}$$

For $\hat{p} = \ln\left(\frac{\epsilon_{rh}}{\epsilon_h}\right) / \ln(r)$

$$\Delta \hat{p} = \frac{1}{\ln(r)} \sqrt{\left(\frac{\Delta \epsilon_h}{\epsilon_h}\right)^2 + \left(\frac{\Delta \epsilon_{rh}}{\epsilon_{rh}}\right)^2}$$

Chaotic regimes?



$$\tau_{MMS} < \tau_{Lyapunov}$$